UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Ingegneria dell'Informazione

Master's Degree Course in Bioengineering

Deep learning applied to 2D video data for the estimation of clamp reaction forces acting on running prosthetic feet and experimental validation after bench and track tests

Thesis Advisor

Prof. Nicola Petrone

Candidate

Samuele Tarabotti

Thesis Co-Advisors

Eng. Mattia Scapinello

Eng. Samuel Cognolato

Academic Year 2022/2023

Abstract

Carbon fiber Running Specific Prostheses (RSPs) have allowed athletes with lower extremity amputations to recover their functional capability of running. RSPs are designed to replicate the spring-like nature of biological legs: they are passive components that mimic the tendons elastic potential energy storage and release during ground contact.

The knowledge of loads acting on the prosthesis is crucial for evaluating athletes' running technique, prevent injuries and designing Running Prosthetic Feet (RPF).

The aim of the present work is to investigate a method to estimate forces acting on a RPF based on its geometrical configuration. Firstly, the use of kinematic data acquired with 2D videos was assessed, to understand if they can be a good approximation to the golden standard represented by motion capture (MOCAP). This was done by evaluating steps acquired during two running sessions (OS1 and OS3) with elite paralympic athletes. Then, the problem was formulated using a deep learning approach, training a neural network over data collected from *in vitro* bench tests, carried out on a hydraulic test bench. Two models were built: the first one was trained over data from standard procedures and validated on two steps of OS1; then, in order to improve the performance of the prototype, a second model was built and trained with data from newly studied procedures. It was validated on three steps from OS3.

Summary

Chapter 1: Introduction

1.1 HISTORY OF PROTHESES

The inability to walk or move properly has been, since ancient times, an heavy burden for everyone who was affected by the loss of a lower limb (Hobara, 2014). Still, men and women have tried to help these people by creating artificial substitutes for the missing legs or feet. The earliest example of prosthesis ever discovered is a wooden toe belonging to an Egyptian noblewoman, dated 950-710 B.C.E.

Figure 1.1. Egyptian toe, first example of a prosthesis ever discovered.

Prostheses were meant to hide the missing leg and, most importantly, to make the amputee able to walk again comfortably. For this matter, the design of artificial lower limbs has evolved in a direction that pursues functionality and comfort for the amputee. The most important progress in these fields was seen during the American Civil War, where the demand for prostheses terribly rose. In this period of humongous loss of limbs and lives, Hanger Inc. was born: it was a company founded by James Edward Hanger (1843-1912), an American engineer whose leg was amputated during the war. He spent his convalescence designing a prosthetic leg

that would later go down in history as the “Hanger Limb”. His company is, to this day, leader in the industry.

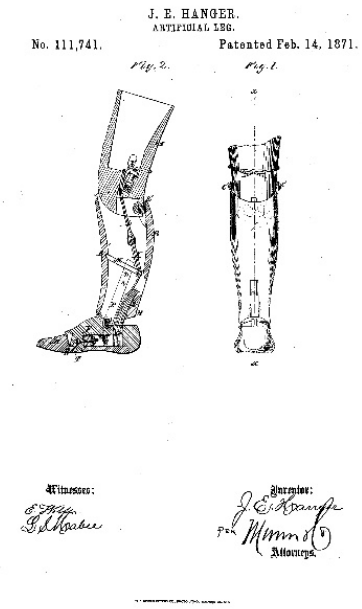


Figure 1.2. The “Hanger Limb”, developed by James Hanger and patented in 1873.

Despite the terrible consequences of the World Wars, the prosthetics field did not develop accordingly. In the late 1950s, the Solid Ankle and Cushion Heel (SACH) model was invented and set the standard for prosthetic design. For the next 20-30 years prosthesis’ design and materials would remain approximately the same until, in the 1980s, advances in composite materials tremendously improved the development of the industry. In particular, carbon composite materials brought substantial improvements in the lightness and durability of prosthetic components such as feet, pylons, and sockets. In 1984 Van Philips, an American inventor of prostheses, created the “Flex Foot®”: the carbon graphite of which it was made allowed the elastic energy to be stored and released during the ground contact phase of the gait.

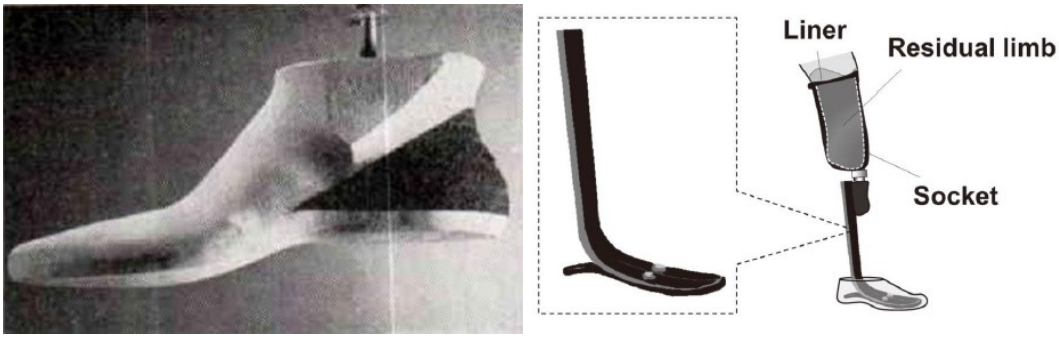


Figure 1.3. SACH foot (left) and schematics of the Flex Foot (right).

The Flex-Foot was the first Running Specific Prosthesis (RSP) and made its appearance in elite sport during the 1988 Paralympic Games. Four years later, the heel component was absent in some athletes, creating the first sprint prostheses. This change in design was first seen in the Flex Sprinter I designed by Össur (Reykjavik, Iceland), the first specialized RSP. Also, the stiffness configuration was changed with the layup sequence of carbon fibers while still maintaining the J-shape outline of the carbon forefoot.

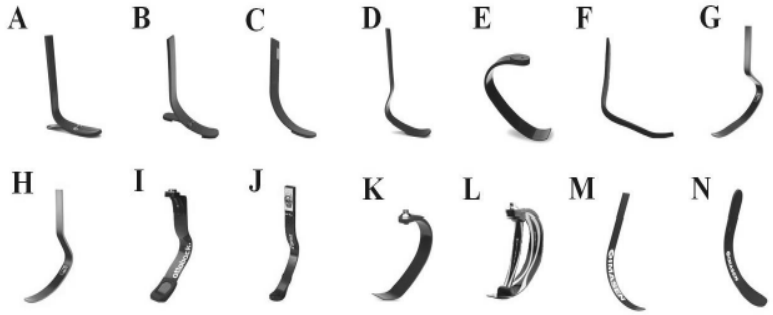


Figure 1.4. A: Flex-Foot® (Modular III; Össur), B: Flex-Sprint II (Össur), C: Flex-Sprint I (Össur), D: Flex-Sprint III® (Cheetah; Össur), E: Flex-RunTM (Össur), F: Symes-Sprint (Össur), G: Cheetah Xtreme®, H: Cheetah Xtend®, I:1E90 (Sprinter, Ottobock), J: 1C2 (C-Sprint®), K: Nitro (Freedom Innovation), L: CatapultTM (Freedom Innovation), M: SP1100 (KATANA, IMASEN Engineering Corp), N: Rabbit (IMASEN Engineering Corp).

There are now several different sprint foot designs available. These new carbon fibers prostheses allow people with lower amputation to walk, run and jump as they could never imagine before. There are now many amputee athletes who can join competitions with able bodied athletes. The latest advancements are strictly connected with the new technologies, for example microprocessors knees allow a prosthetic to adapt its flexion and extension for

different environments. Moreover, 3D printed prosthetics have sparked a renaissance of cost-effective DIY prosthetic design and production.

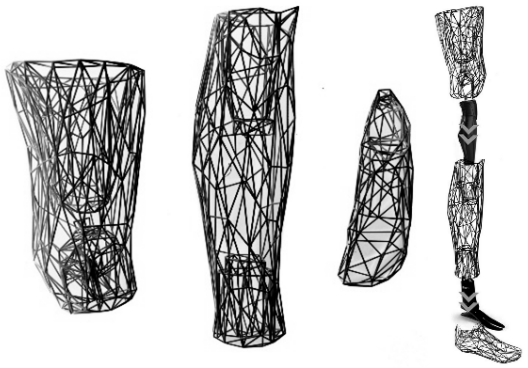


Figure 1.5. 3D-printed titanium limb.

1.2 RSP COMPONENTS

As seen in figure 1.6, running specific prostheses are built out of various components. The main ones are listed below:

- **Socket**: the portion of the prosthesis that fits around and envelopes the athlete's stump and to which the connective components are attached. In later years, sockets have been often obtained with 3D printing techniques, in order to ensure maximum compatibility with the athlete's residual limb and to make the system lighter.
- **Connecting components**: they can be fixed for transtibial amputees or with some degrees of freedom for transfemoral athletes (for example, the prosthetic knee).
- **Sport Knee Joint***:* It is a cylindrical, hydraulic joint that connects the socket to the RPF in transfemoral amputee. The lower leg of the prosthesis swings through freely to an angle of about 60° in the flexion direction. Even at high stride rates, a smooth extension stop is ensured by extension damping applied along the entire extension movement, which increases significantly shortly before reaching the end position. Flexion and extension damping can be adjusted separately and individually.
- **Running prosthetic foot**: carbon-fibre blade.
- **Spiked sole**: Sole that is used during sprinting and jumping competitions, that provides a specialized traction that has been optimized for the blade. The spikes ensure grip with the Tartan surface, in which the athletics tracks are made.

Figure 1.6. RSPs main components.

There are two main types of connecting components:

- **Ottobock® pyramid attack**: the pyramidal attack consists of a pyramid receiver and a pyramid adapter in titanium. This system allows to quickly change the foot and allows adjustment of the angles of alignment in the sagittal plane and in the frontal plane, acting on the four screws. This method is usually implemented for C-shaped feet.
- **Direct connection** of the blade to the socket by screws: this method is only applied on J-shaped prostheses. It is lighter because no adapter is necessary, but less adjustments can be made: sometimes angular wedges are put between the socket and the J-foot to vary the inclination angle in the sagittal plane.

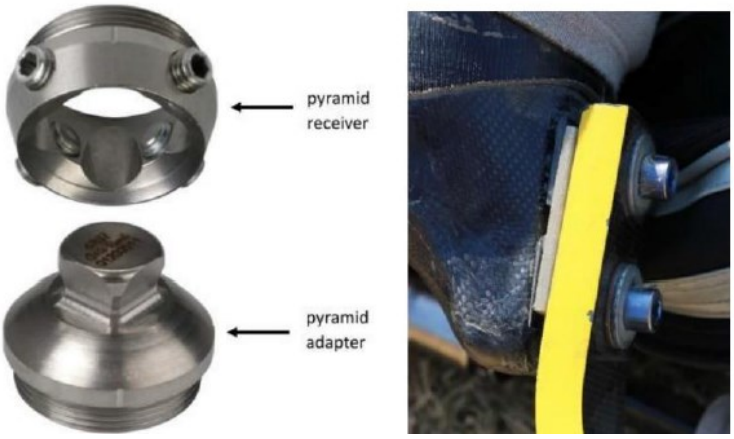


Figure 1.7. Pyramidal attack (left) and fixing screws (right).

1.3 BIOMECHANICS OF RUNNING WITH A RSP

RSPs are designed to replicate the spring-like behavior of biological legs. To describe this, the whole body is often modelled with a spring-mass model: it consists of a massless spring and a particle that represents the athlete's center of mass. During the first part of the stance phase, the compressed spring stores potential energy, which is given back with a certain efficiency in the second part of the stance phase.

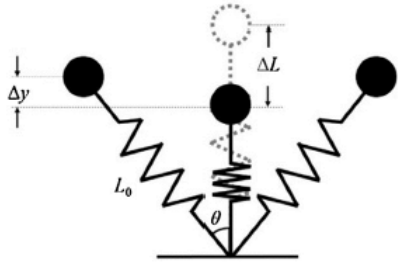


Figure 1.8. Spring mass-model schematization of a running gait.

Using this model, leg stiffness is defined as the ratio between the maximum vertical Ground Reaction Force and the maximum leg compression (Sano et al., 2017).

$$K_{leg} = \frac{vGRF_{peak}}{\Delta L}$$

To be able to consider different athlete's weight, the leg stiffness can be made dimensionless by multiplying it by the ratio of L_0 and body weight BW.

$$K_{leg} = k_{leg}\left(\frac{BW}{L_0}\right)$$

It has been demonstrated that this parameter remains constant over a wide range of speeds. ΔL is calculated following Farley and using the length of the leg spring at the touchdown, which was estimated to be equal to the standing leg length L_0, and half of the angle swept by the leg spring while the foot was on the ground (ϑ).

$$\Delta L = \Delta y + L_0(1 - cos\vartheta)$$

This linear spring model doesn't exactly replicate the behavior of biological legs: in fact, the structure of human lower limbs is viscoelastic, and only a portion of the potential energy stored can be returned. The mechanical energy lost is the hysteresis cycle, needed in order to keep a constant speed during running, is provided by muscular force. The magnitude of said additional force is inversely related to leg stiffness and is influenced by the magnitude and orientation of the external forces.

As previously explained, RSPs are made by carbon fiber composites materials that are characterized by great energy storing capabilities. Therefore, this material can recreate the tendons' storage of mechanical energy and its return during ground contact. The main difference with biological legs is that ankles can generate mechanical power while RSPs cannot, and so the energy that is given back during a RSP running step goes only from 63% to 95%. Moreover, prosthesis stiffness cannot change during running. Stiffness and other parameters such as the length of leg or the vertical landing velocity directly influences the contact time and, consequently, the stride frequency of the steps. Thus, RSP's stiffness is a parameter that must be carefully designed in order to increase the athletes' performances.

Y. Sano et al. demonstrated that, in *transfemoral* amputees, $Kleg$ of the prosthetic leg was approximately 12% smaller than that of the intact leg. These results are congruent with previous finding demonstrating that *transtibial* amputees wearing RSP have bilateral asymmetry in $Kleg$ during running: $Kleg$ in transtibial amputee sprinters remained constant or increased with speed in intact limbs, while it remained constant or decreased in limbs using RSPs.

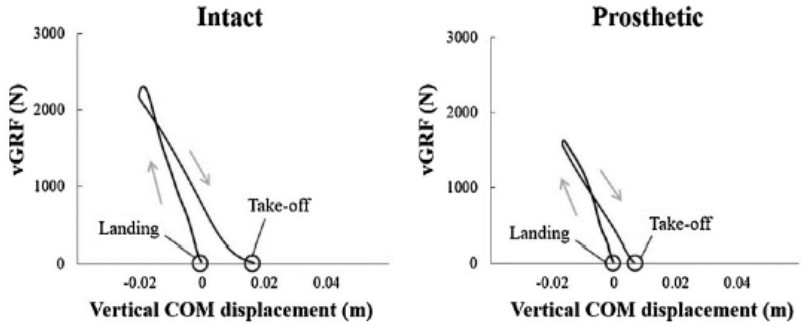


Figure 1.9. Typical examples of vGRF-COM displacement curves recorded from one subject for the intact (left) and prosthetic leg (right). Note the bilateral differences in vGRF. (Sano et al., 2017)

These studies, along with the previous data, suggest that bilateral asymmetry in spring-like leg behavior may be a common biomechanical characteristic among lower extremity amputees wearing the RSPs while running. Although Y. Sano and al. observed that there was no difference in ΔL, $vGRFpeak$ was significantly greater in the intact leg than in the prosthetic leg. These results indicate that asymmetric spring-like leg behavior in *transfemoral* amputees with RSP is mainly due to the bilateral differences in $vGRF$.

1.4 RUNNING GAIT ANALYSIS TERMINOLOGY

Gait analysis is the general framework for this project. For a better understanding of the following chapters, here is proposed a summary of the main concepts of running gait analysis.

One of the fundamental concepts of gait analysis is the *gait cycle,* which is the period from initial contact of one foot to the next initial contact of the same foot (Dugan & Bhat, 2005). In normal walking, there are two phases of gait – *stance* and *swing*. During one gait cycle in walking, stance phase represents 60% of the cycle while swing phase represents the remaining 40%. *Double support*, when two limbs are in contact with ground, occurs during the first and last 10% of a particular stance phase. *Single limb support* is equal to the swing time of the opposite limb.

The running gait cycle, instead, can be divided into stance phase, swing phase, and *float phase*. The first half of the stance phase is concerned with force absorption, whereas the second half is responsible for propulsion. Stance phase can be divided into subphases of initial contact (also called *heel strike*) to midstance, and midstance to toe-off. To understand the biomechanical events during running, stance phase can be furtherly divided into three major components:

initial contact to foot flat, foot flat to heel-off, and heel-off to toe-off. Swing phase during running can be divided into initial swing and terminal swing; float phase occurs at the beginning of initial swing and the end of terminal swing.

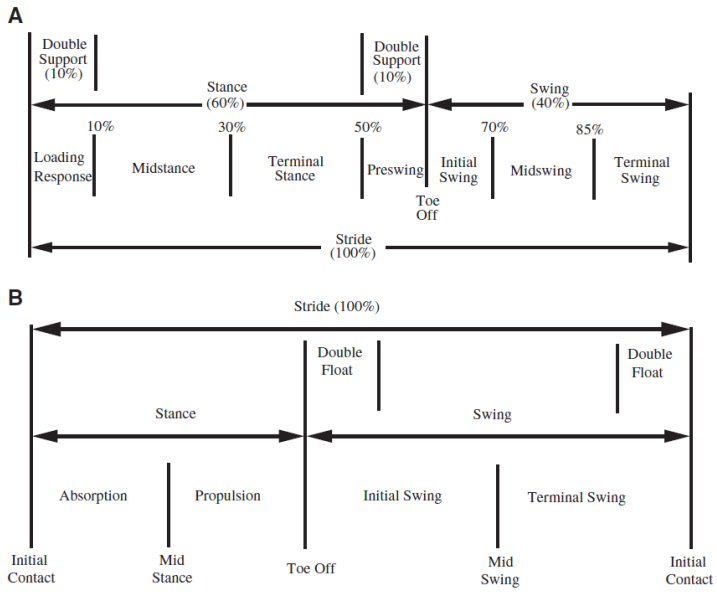


Figure 1.10. Schematic representation of walking gait (A) and running gait (B).

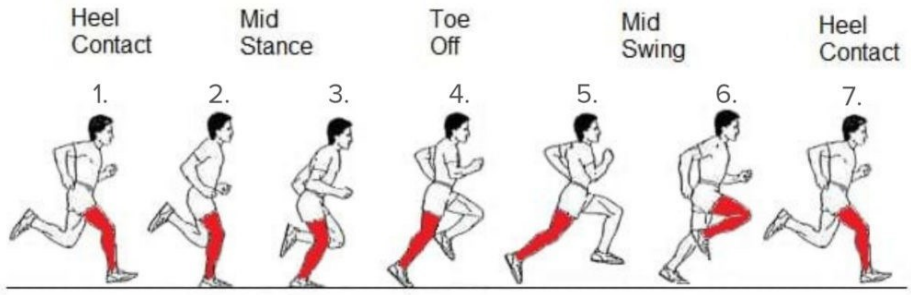


Figure 1.11. Visual representation of running gait (Dugan & Bhat, 2005).

The terminology explained in this section, along with the overview reported in the chapter, are meant to serve as an introduction to the work carried out during this thesis in the estimation of forces on *Running Prosthetic Feet* (RPF), as described in detail in the following chapters.

Chapter 2: Context of the work

2.1 PROJECT OLYMPIA

This thesis was done at University of Padua in collaboration with Centro Protesi INAIL (Istituto Nazionale per l'Assicurazione contro gli Infortuni sul Lavoro) in Vigorso di Budrio, as a part of project Olympia. Said project has the objective of developing innovative technologic solutions to enhance paralympic athletes' performances and safety during sports competitions and training. In particular, the research poses its focus on prostheses and orthoses designed for running and jumping, and on prostheses and monoskis used in alpine skiing.

The work described in this paper was carried out with the aim of trying to develop a simple and standardized method which uses deep learning for the evaluation of the forces acting on a running prosthetic foot based on video acquisitions of track tests and performances.

2.2 RELATED LITERATURE AND EARLIER THESIS WORK

Over the years, many students at University of Padua have worked for project Olympia as part of their thesis work.

For his thesis project, Gianfabio Costa (Costa, 2018) developed an hydraulic test bench able to reproduce as closely as possible the biomechanics of the running, in order to evaluate in detail and in a complete way the mechanical performances of RPF. Said test bench, named Colossus, will be described in detail later.

Gianmario Foscan (Foscan, 2018) developed and validated a method to apply strain gauges on RPFs to evaluate ground reaction forces and loads on the sockets during in-vivo applications, without the need of dedicated load cells or force platforms.

Antonio Gri (Gri, 2019) evaluated the effect of different socket alignments on the performance of elite Paralympic athletes during running and long jump thanks to an instrumented running prosthetic foot.

Leonardo Bonato (Bonato, 2020) used Colossus test bench to create a method to characterize

prostheses' behavior in terms of stiffness and studied its variation during a step. He discovered that the main parameters to consider are the orientation of the foot to the ground (θ_G) and the combined effect of the two forces acting in the sagittal plane (ρ=Fx/Fy). He also developed a mathematical model called Virtual Stiffness Map that, from a reasonable number of tests, can determine the elastic behavior of an RPF starting from the θ_G and ρ values and validated it on an in-vivo test.

Alessandro Luisi (Luisi, 2020) defined a method for developing 3D FEM models of running specific prostheses, through which it was possible to predict and study the mechanical behavior of the foot for different constraint and load conditions.

Taking further Luisi's work, Antonio Martellato (Martellato, 2022) carried out a mechanical characterization of a specific foot and developed a finite elements model that could simulate its behavior both during tests on Costa's Colossus and during in-vivo applications. The model was able to successfully predict the load response of the foot during in-vitro conditions not tested on the Colossus and to predict the load response of the RPF based solely on its cinematics, acquired from an in-vivo test.

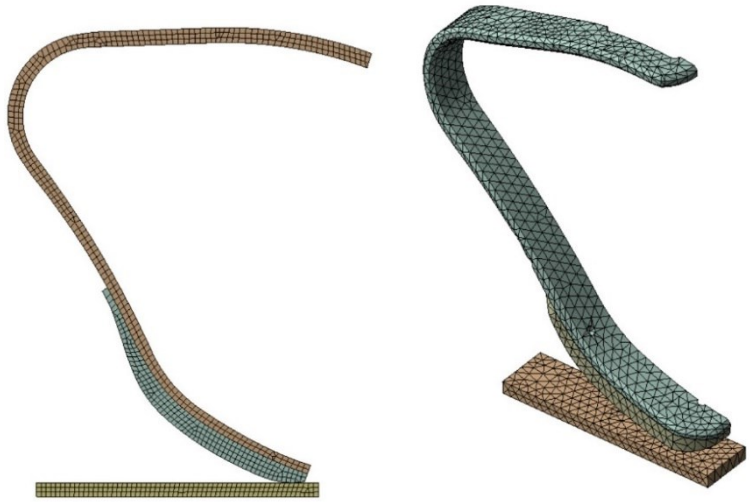


Figure 2.1. 2D (left) and 3D (right) FEM models of Ottobock Runner Standard CAT 4 developed by Antonio Martellato.

There are also many works of scientific literature that try to investigate RSPs' behavior. For example, Shepherd et al. (Shepherd et al., 2019) proposed a simple custom FEM model developed on MATLAB that uses a linear correlation between forces and displacements. Petrone et al. (Petrone et al., 2020) developed two wearable instrumented running prosthetic feet starting from commercial RPFs, suitable for load data collection on the track thanks to strain

gauge bridges, that is are able to transmit data via Wi-Fi to a base station for real-time monitoring.

Beck et al. (Beck et al., 2016) evaluated the stiffness of a great number of RPF of different models and categories, in order to have a more accurate classification following performance based metrics instead of subjective stiffness, as it is currently recommended by prosthetic manufacturers.

Migliore et al. (Migliore et al., 2021) carried out an explorative study on an elite transfemoral Paralympic athlete to test the response to an innovative alignment based on able-bodied athletes. The new alignment allowed the prosthetic side hip to work in a more comfortable range, increase the horizontal propulsion, decrease the running effort, avoid knee buckling, while not increasing the braking forces at the sound side. Moreover, the biomechanical evidence appeared to be highly related to the subjective perceptions of the elite athlete tested in the study.

Figure 2.2. Instrumented Ottobock Runner CAT 4, developed by Petrone et al.

2.3 AIM AND STRUCTURE OF THIS WORK

The work presented in this paper aims to investigate whether it is possible to estimate the clamp reaction forces acting on a running prosthetic foot with the sole use of kinematic data obtained with 2D videos.

This is done by preliminarily understanding if videos are a reliable source of kinematic data by comparing them with the current golden standard, that is optical motion capture. This first task was accomplished by acquiring both video and motion capture data of the same track runs and comparing them.

The second and most important objective was to find a method to univocally correlate the kinematic configuration of a RPF and the clamp forces acting on it at a given moment. This has been done by carrying out some preliminary analysis to understand if the problem could be solved with deterministic methods; then, a deep learning approach was deemed to be more optimal. The latter was applied by building and training a Neural Network that, taking the coordinates of the markers on the RPF as input, could predict the forces acting on it.

In chapter 3, the theoretical background of deep learning is reported, as well as the description of the type of model used.

Chapter 4 describes the test bench used to collect training data for the network.

In chapter 5 a running session to acquire *in vivo* data is reported; moreover, the confrontation between video and MOCAP data is described.

Chapter 6 reports the standard procedure for *in vitro* tests, the preliminary analysis, and the construction and validation of a first neural network.

Finally, chapter 7 is concerned with the study of new *in vitro* testing procedures for the training of an improved model. The validation of said model is then described.

Figure 2.3 represents a descriptive flowchart of the work.

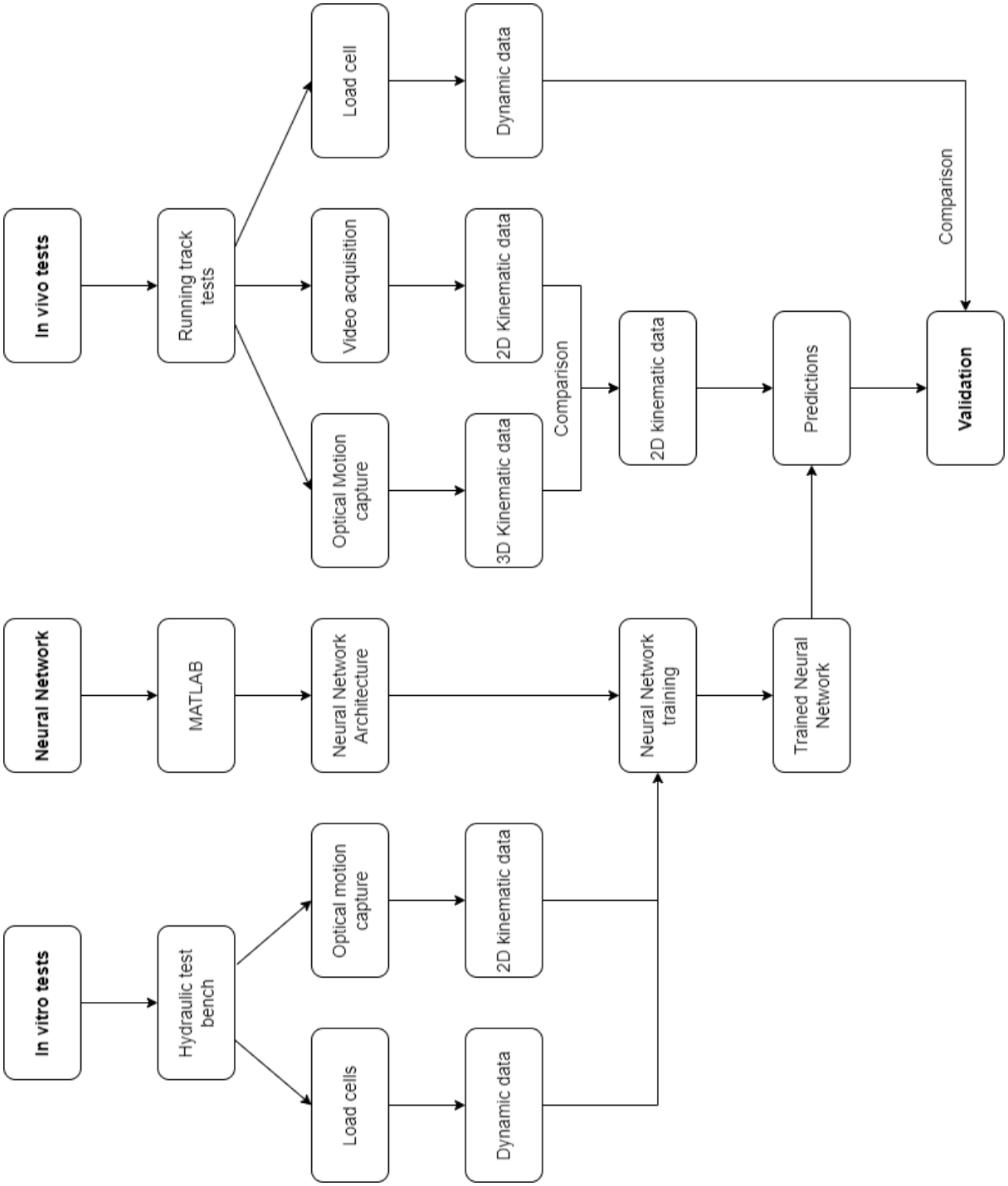


Figure 2.3. Flowchart of this thesis.

Chapter 3: Introduction to deep learning theory

3.1 INTRODUCTION TO DEEP LEARNING

In recent years, deep learning has become ubiquitous in the world of scientific research thanks to its wide range of applications (Razavi, 2021). Its minimalistic approach requires the sole use of big quantities of data to create accurate predictive models, which is possible thanks to the boom in computational power and the emergence of big data sources, associated data storage, and sharing technologies. Recent remarkable applications include unprecedented performance in various fields, such as image generation and processing, image recognition, natural language processing, sequence prediction, and reinforcement learning.

Specifically, Deep Learning (DL) is a branch of *Machine Learning* (ML), which falls under the umbrella of the field of *Artificial Intelligence* (AI). ML is nowadays concerned with developing machines that improve their own performance in carrying out a given task over time by "learning" from examples, with minimal human efforts to instruct the machines how to do so. However, Deep Learning techniques can be applied simply with limited expertise of ML, making them very accessible and versatile: DL frameworks are now available for most widely known scripting and programming languages, such as the Pytorch framework for Python and the Deep Learning Toolbox for MATLAB.

The defining models of Deep Learning are Artificial Neural Networks (ANN), which, by applying many possible algorithms, recognize underlying relationships in a set of data. The intuitive functioning of these models is that after they've been trained with a sufficient amount of data, which in the supervised learning case (as explained in section 2.2) is intended as pairs of inputs and their respective outputs, they can accurately predict the output for unseen examples. The algorithms that allow this behavior will be explained in the case of the Multilayer Perceptron, which is the model used in this work.

A fundamental property of ANNs is that they have been proven to be *universal approximators* (Hornik et al., 1989). This proof indicates that even basic Neural Networks would be able to

approximate any function with any desired level of accuracy, provided that quantity of available training data and complexity of the model match the complexity of the problem. This is especially important because it allows DL models to be applied to problems with an enormous number of variables, which would be too complex to be identified and processed through purely deterministic methods.

Since this groundbreaking discovery in the late 1980s, the "universal function approximation theorem" has been the fundamental driver of interest in Neural Networks across a variety of disciplines and applications, as stated before.

3.2 STATISTICAL LEARNING

Statistical learning (SL) is the general framework for machine learning, drawing from the fields of statistics and functional analysis (Cognolato & Testolin, 2021) (Razavi, 2021). SL can be both *supervised* and *unsupervised*: unsupervised learning refers to a process where a model learns some kind of structure in the data from "unlabeled" examples, which are inputs with no associated output; this is as opposed to supervised learning where the objective is to predict some quantity from an input set of data. Here examples are "labeled", meaning the output associated with each input is known. In this work, only supervised learning models have been used.

When dealing with a supervised problem, the purpose of statistical learning can be formulated as follows: based on a given set of observations $S = \{(x_1, y_1), (x_2,y_2), \ldots, (x_n, y_n)\}$, the aim is to find a model $y = f(x)$ that approximates an unknown function $y \sim F(x)$, with $x \in X$ and $y \in Y$ sampled with probability $p(x,y)$ from an unknown data distribution D defined over the possible inputs and outputs of X and Y. A *loss function* l can be defined as the measure of the distance between the model's prediction and the ground truth sampled from D. l is also called *unreduced loss function*, because no operation of averaging has been performed on it. The model has to minimize the *expected risk* $E_D[l]$, where $E[\cdot]$ is the expected value, defined over the entire distribution D. Since such quantity is unknown, the model must rely on a proxy measure defined only on the dataset S, called *empirical risk*:

$$L(S) = E_s[l] = \frac{1}{n}\sum_{i=1}^{n} l(f(x_i), y_i)$$

$E_s[l]$ is the expected value of the loss over the observations in the dataset. $L(S)$ is the *reduced loss function*, which from this point onwards will be referenced as *loss function* or just *loss* for more concise language.

The scope is to find a fitting model f through a *search procedure*. In the case of Artificial Neural Networks, the model will be found through a learning procedure.

DL (and ML in general) can be used to approach a huge range of complex applications, but most of them consist in solving the combination of two types of problems: *regression* and *classification*. The main goal of regression is to estimate a continuous mapping function with input and output values, while classifications is a predictive model that approximates a mapping function from input variables to identify discrete output variables, which can be classes or categories. As explained further in section 3.5, The problem considered in this paper is a typical regression problem, where an ANN tries to estimate the clamp reaction forces on a RPF based on the configuration of the markers placed on it.
Since statistics offers limited solutions to complex regression problems such as the estimation of non-linear functions, DL can be considered as a tool to further investigate problems that would be too difficult to solve analytically.

3.3 MULTILAYER PERCEPTRON

Artificial Neural Networks are a family of statistical learning models whose functioning intentionally resembles the one of neurons. The base operative unit of every ANN is in fact called *neuron*: similarly to how brain cells work in their context of the body, ANN's neurons constitute a simple linear model which produce a positive or negative output given the input x and a set of weights w:

$$f(x, w) = \sum_{i=1}^{n} x_i w_i$$

They work through the layering of linear combinations followed by non-linear functions. The most basic model is the *Perceptron*, developed by Frank Rosenblatt in 1958. It is a binary classifier taking an input vector x and computing the output class y through a simple linear relationship:

$$y = f(Wx + b) = \begin{cases} 1 & if\ Wx + b\ >\ 0 \\ 0 & otherwise. \end{cases}$$

W and b are learnable parameters called respectively *weights* and *biases*. They can be fine-tuned following an optimization algorithm that minimizes the empirical risk. The function f is called *activation function* and, in a generic model, can be any non-linear function. In the case of the Perceptron f corresponds to the *Heaviside* function, also called "step function". So, at its core, the Perceptron corresponds to a linear regression model usable for both regression and classification. By stacking a higher number of layers and using more complex activation functions, the model can be generalized to a *Multilayer Perceptron* and can be used to approximate highly non-linear functions.

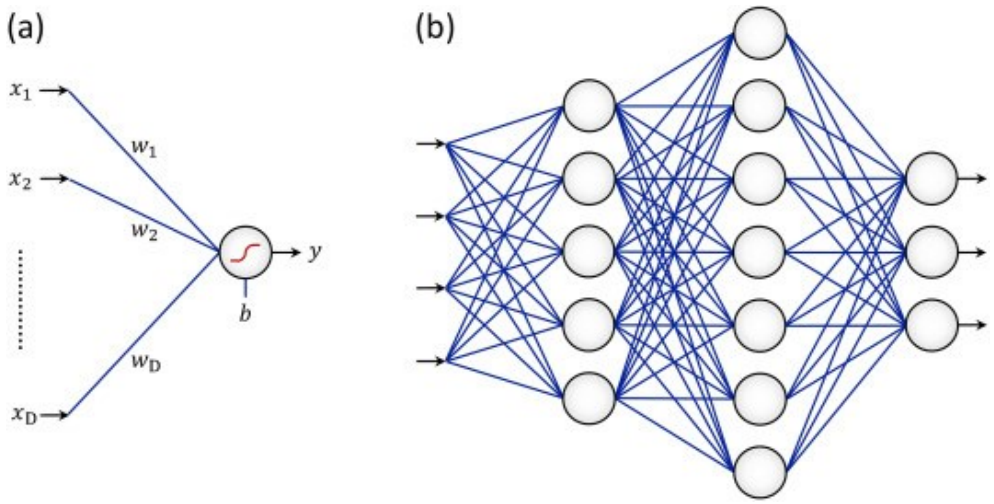


Figure 3.1 Perceptron (a) and Multilayer Perceptron (b).

The Multilayer Perceptron belongs to the category of Feedforward Neural Network models. These types of Neural Networks are named after the *feedforward path*, where data is progressively transformed in a forward fashion. The structure of Multilayer Perceptron, as illustrated in figure 3.2, is the following:

- Input layer: the first layer, where the input data x is received by the Neural Network. From this point, data is transformed through the hidden layers in order to compute the output.
- Hidden layer(s): hidden layers are optional intermediate layers which increase the complexity and the depth of the network. Each hidden layer composed of neurons, as shown in fig. 3.1, receives the output from the previous hidden or input layer and returns the input for the next hidden or output layer.

- Output layer: the last layer, where the output y is computed.

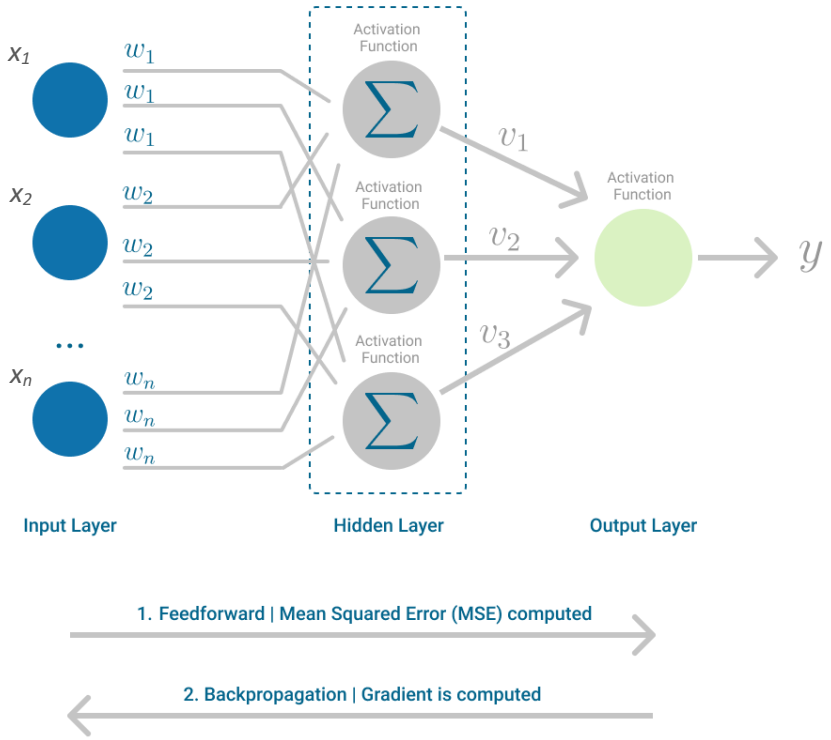


Figure 3.2 Architecture of a generic Multilayer Perceptron with one hidden layer and n neurons (Carolina Bento, 2021). The computation of the output prediction y is the result of the feedforward path.

3.3.1 Feedforward path

The mechanism that enables the Multilayer Perceptron to actually learn is the combination and iteration of two processes: the *feedforward path* and the *backpropagation*. A single iteration of the process over the whole dataset takes the name of *epoch*.

The feedforward path is the path followed by the input data that results in the output. The input x is received by the input layer and linearly combined with weights and biases for every neuron in the first hidden layer. The result of this operation, called *linear activation*, has to pass through the activation function, which returns the output of the hidden layer. The process is repeated for every hidden layer. The computational process of an hidden layer takes the name *forward pass*: the feedforward path is the composition of every layer' forward pass. The output layer operates a dimensionality change in order to fit the number of dimensions of the output. Then, the loss function is finally computed, measuring the distance between the network predictions y and the actual data $\bar{y}$, the ground truth. Different loss functions can be used, but in simple

regression applications such as the one seen in the present work, the *Mean Square Error* (MSE) is the most applied. For two generic vectors $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^n$, the MSE is the L^2 norm of the difference between the two vectors. It is computed as follows:

$$MSE(a, b) = \frac{1}{d}\sum_{i=1}^{d}(a_i - b_i)^2$$

The MSE is the result of the feedforward path and the point from which backpropagation starts.

3.3.2 Backpropagation

Backpropagation is the learning mechanism that allows the Multilayer Perceptron to iteratively adjust the weights in the network, with the goal of minimizing the loss function. All the main techniques used in the state of the art today are based on an iterative method called *Gradient Descent* (GD). The main idea is to move the parameters (weights and biases) in the direction where the loss function L decreases, which is the one given by the negative gradient of L with respect to the parameters, computed on the predictions and ground truth on the dataset S. In each iteration, after the weighted sums are forwarded through all layers, the gradient of the MSE is computed across all input and output pairs. Then, to propagate it back, the weights of the first hidden layer are updated with the value of the gradient. This way, the weights' gradient is "backpropagated" to the starting point of the neural network. The partial derivatives of the loss function with respect to each layer' parameters are computed through the *chain rule of derivation*.

In general, the gradient iteration at step t+1 is computed with the following update:

$$\Delta w_{t+1} = \Delta w_t - \eta L(w_t)$$

η is a hyperparameter known as *learning rate*.

A common improvement that can be made is to use a subset of samples B ⊂ S instead of the entire dataset, called *batch* or *minibatch*. This improves computational performances and introduces noise which may be beneficial for the learning process by approximating the full gradient with a partial gradient, noisy due to the sampling of B ⊂ S. The new update rule takes the name of *Stochastic Gradient Descent* (SGD).

Common alternatives to SGD are its *momentum* variant (Rumelhart et al., 1986) and the *Adam* optimizer (Kingma & Ba, 2017). SGD with momentum takes inspiration from physical

momentum: a fraction of the "movement" from the previous iteration is retained in the next one. The parameters at iteration t+1 are thus:

$$\Delta w_{t+1} = \Delta w_t - \eta L(w_t) + \alpha \Delta w$$

This technique allows to get over local minima, where regular SGD would otherwise get stuck. Adam is a state-of-the-art optimizer which computes the running averages of both the first and second moments of the gradient, being respectively the mean and the uncentered variance. The update rule is then:

$$w_{t+1} = w_t - \eta \frac{m}{\sqrt{v} + \epsilon}$$

where m and v are the unbiased first and second moments. ϵ is typically a very small scalar. The advantage of using Adam over SGD is that it combines two useful characteristics: firstly, it maintains a per-parameter learning rate that improves performance on problems with sparse gradients; secondly, it also maintains a per-parameter learning rate component that is adapted based on the magnitude of change in the weight, improving the solution of problems with high amounts of noise.

Since the update rule requires the derivative of the error regardless of the optimizer, a hard requirement for every Neural Network is that the activation function in every layers must be differentiable.
The optimizing process goes on until the network reaches satisfactory performances on the problem, where the model is considered converged.

3.3.3 Activation functions

For a better understanding of the training process, it is important to briefly describe the typical activation functions used for regression problems. As previously stated, an activation function can be any non-linear function that takes the linear activation as input. The most common functions are:

- **Sigmoid**: is an element-wise function that produces values in the interval [0, 1].

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

The sigmoid was used as the main activation function for neural networks together with the hyperbolic tangent tanh(x), but both have been replaced in most uses by rectified linear units

(ReLUs) and its variants. Despite the success of ReLUs, both functions still maintain some uses in specific applications: for example, the sigmoid function is widely used in the output layer of binary classificators.

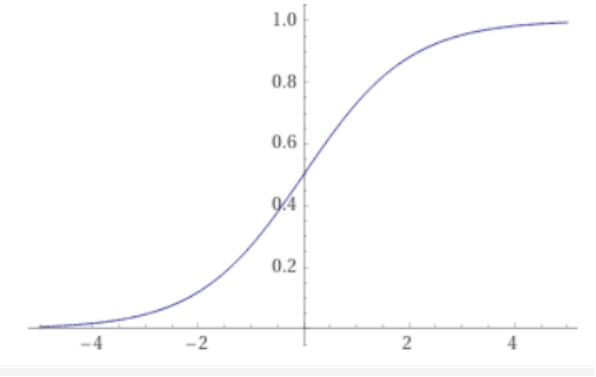


Figure 3.3 Sigmoid.

- **Hyperbolic tangent**: is an element-wise function that produces values in the interval [-1, 1].

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

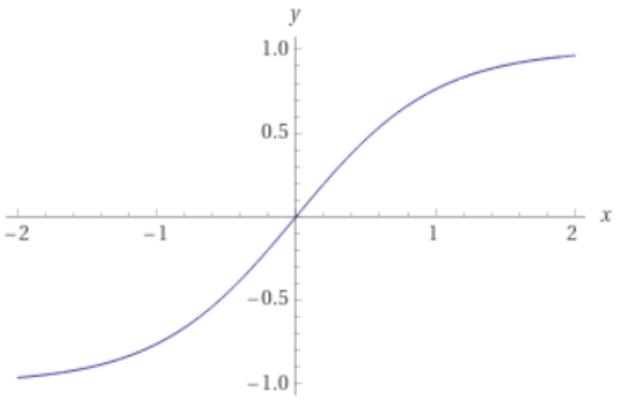


Figure 3.4 Hyperbolic tangent.

- **Rectified Linear Unit (ReLU)**: is an element-wise function which outputs the input if positive, otherwise it outputs zero.

$$ReLU(x) = x^{+} = max(x, 0)$$

It is a simple yet effective function that's very easy and versatile both in the feedforward path and in the backpropagation. Due to its simplicity, it has replaced the sigmoid and the hyperbolic tangent in the hidden layers of modern neural networks. It is important to notice that since activation function must be differentiable, the derivative in x=0 (where the function is non-differentiable) is set to be null.

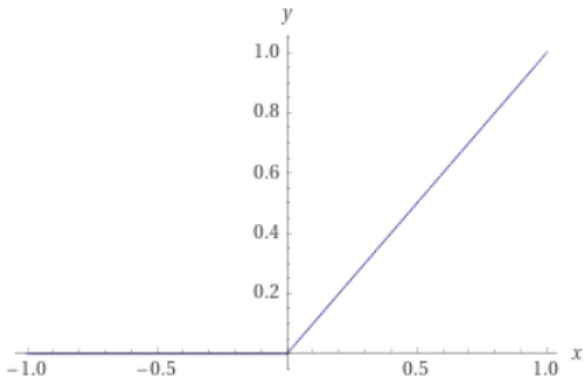


Figure 3.5 Rectified linear unit.

3.4 OVERFITTING

Given the high level of flexibility of ANNs, a common problem that may arise during the training phase is *overfitting*, which refers to a situation where a model fits the noise in the training data rather than the underlying function. Before describing the possible solutions to this problem, it is important to understand how a generic dataset S is typically split in a Neural Network application.

The available data are usually divided into three sets, commonly referred to as *training*, *validation*, and *testing* datasets (Razavi, 2021). Training datasets, as the name suggests, are used for the training of the network, as described previously; testing sets are instead used to provide an unbiased evaluation of a final model fit on the training data set. The validation dataset is needed to leash the hyper-flexibility of the network while training. Validation sets are not mandatory for the technical functioning of the training process; however, their use is advisable in order to prevent overfitting. This set is used to approximate the generalization loss at every iteration, to verify the model's performance on unbiased data. The simultaneous use of training and validation datasets during ANN training may be best described with the *early-stopping* strategy: in this approach, the quality of fit to the validation dataset is evaluated after each training iteration, that is an optimization iteration trying to minimize the loss function on the training data. Empirically speaking, as the training error decreases over time, the validation error decreases as well. However, at some particular epoch, the validation error may begin to increase while the training error keeps decreasing. This epoch marks the beginning of overfitting; thus,

after a set number of iterations with this behavior, the training process is stopped. This strategy is therefore called early stopping in the sense that the training is stopped before it can further improve the fit to the training dataset, which will lead to overfitting.

When the training process stops, the performance of the final model is assessed through the test dataset. In general, it is important for the testing dataset to not include samples from the training set, as having overlapping samples could lead to false conclusions over the accuracy of the trained network.

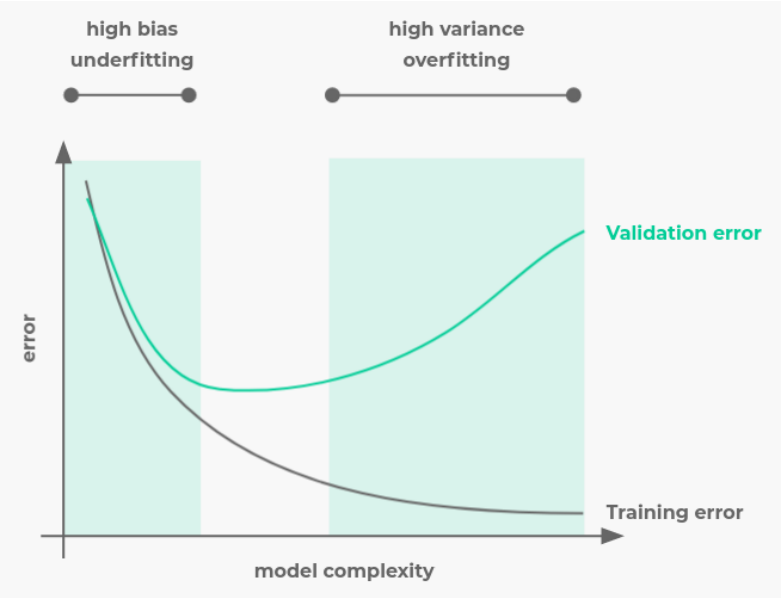


Figure 3.6 Schematic representation of overfitting in ANNs. Overfitting happens when the training distribution fits perfectly on the model, which is not able to generalize to out of distribution samples.

Another commonly used strategy to put a leash on the hyper-flexibility of ANNs is *regularization*. Unlike early stopping, this strategy tries to minimize a *regularization function* during training, which is added to the loss function. A traditional regularization function in the ANN context is the sum of the square of all network parameters, based on the notion that the smaller the parameters of a neuron, the less complex it is. In the context of Deep Learning, this technique takes the name of *weight decay*. The final objective be minimized becomes the sum of the main loss function and the squared L^2 norm:

$$L = L_{main} + \lambda \|p(w,b)\|_2^2$$

The hyperparameter λ is called *regularization coefficient* and determines a tradeoff between the main loss function and the regularization term. In general, each regularization term is weighted by a coefficient, and the best values are found through hyperparameters searching.

3.5 APPLICATION TO THE ESTIMATION OF FORCES

Having explained the theoretical background to DL, it is now possible to formulate more precisely the problem considered in the present work. The rigorous description of the architecture of the network, the training process, and the results of the analysis are found in chapter 6 and 7.

The aim is to build a Neural Network that takes the coordinates of the markers placed on a prosthesis as input and computes the clamp reaction forces acting on it as output. As anticipated, the model used is a simple Multilayer Perceptron with $2n + 1$ inputs and 2 outputs, n being the number of markers on the prothesis: for each marker, an (x, y) couple of coordinates is used. The additional input parameter is represented by the ground angle θ. The two outputs correspond to the reaction forces F_x and F_y in the clamp coordinate system, often referenced, respectively, as F_2 and F_1.

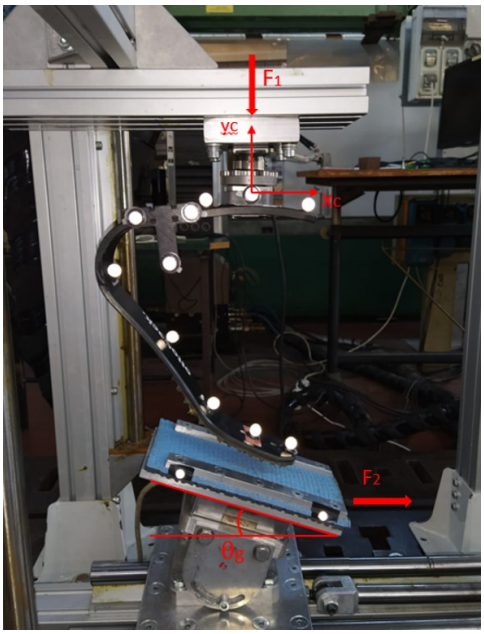


Figure 3.7 Graphic visualization of the variables of the network.

The dataset for the training and testing of the network is carried out considering input-output couples acquired during tests on Colossus hydraulic test bench: the markers' configuration is acquired with optical motion capture technology, while the forces are acquired through load cells.

Chapter 4: Hydraulic bench for RSP testing

4.1 TEST BENCH COLOSSUS MACHINE

Colossus test bench was designed by Gianfabio Costa in a previous work of thesis (Costa, 2017). The aim of the test bench is to determine the RSPs stiffness, applying a known vertical force and measuring the corresponding reaction forces. In this work, it has been used to collect kinematic and dynamic data to train the Neural Network.

Figure 4.1. Colossus test bench machine.

Two hydraulic actuators are used to apply the forces F1 and F2 and execute the displacements C1 and C2. Cylinders were manufactured by MTS Systems Corporation and can produce a maximum force of 14700 N, by the oil control system composed of MOOG servo-valves. The control software and control station are produced by MTS Systems Corporation as well, which operates feedback control on the servo-valves.

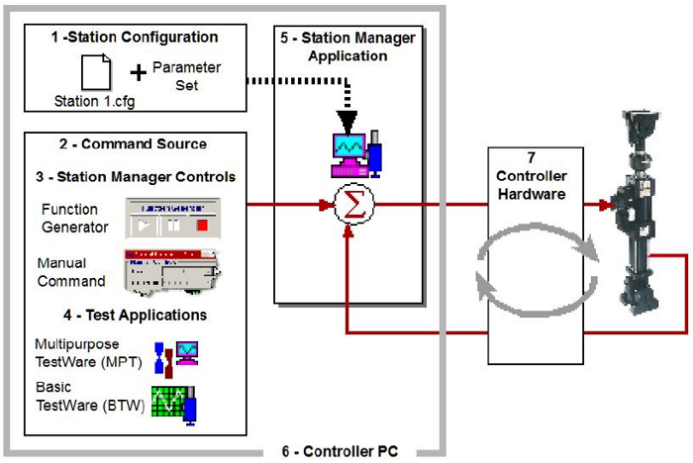


Figure 4.2. Colossus machine control system.

The movement of the pistons can take place under force control with a uniaxial force transducer, placed on the head of the steam or displacement control using a potentiometer inside the cylinder. A double portal has been designed using structural aluminium profiles that hold up the hydraulic actuators. This is a very flexible solution because it allows to easily change the project configuration. The aluminium profiles have a square section with unified T-slot on each side, to allow the insertion of the fixing nuts. The internal slots are designed to minimize weight while maintaining a high moment of inertia of the section. The portal height is about 3 m to have enough space for the vertical actuator, the vertical slide, the prosthesis, and the horizontal slide. The width is instead set to 500 mm, so that the horizontal slide can comfortably accommodate.

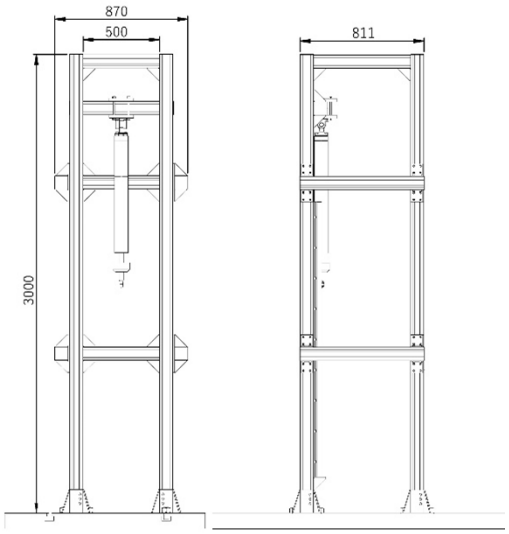


Figure 4.3. Colossus frame and vertical cylinder attachment.

A horizontal beam was placed on the slide, so that the prosthesis could be moved upwards and backwards to align the Centre of Pressure (COP) with the plane identified by the two axes of the rails.

The horizontal slide, made with the same profile of the vertical one, was positioned on two cylindrical bars where it could slide backwards and forward. On the top of the slide was put an aluminium plate which can be covered with Tartan, the material in which athletics tracks are made. If the prosthesis being tested have a rubber sole instead of a spiked one, the aluminium plate is covered with sandpaper to minimize the sliding between the two surfaces. θ_G angles, that will be discussed in detail in the following paragraph, were set by inclining the surface around a hinge. Because of the high loads acting on the RSPs tested, a flange was assured by friction to keep constant a certain angle. Initially, there were two couple of flanges holed at prefixed angles, the first one at -45°, -30°, -15°, 0°, 15°, 30°, 45°, the second one at -37.5°, -22.5°, -7.5°, 7.5°, 22.5°, 37.5°. The flanges are connected to the sledge with screwed pins that are inserted in the holes to replicate the desired θ_G. During the course of this thesis, one more couple of flanges was used, with a continuous fissure. This was done to reproduce a wider array of angles, without being limited to an interval of ±7.5.

Figure 4.4. Horizontal sledge connected to the holed flange to replicate the angles of θ_{SH}.

As seen in the picture below, the system has three degrees of freedom: two translations exerted by the hydraulic actuators and one rotation of the inclined sledge. Obviously, the translational ones are continuous, while the rotational is discrete.

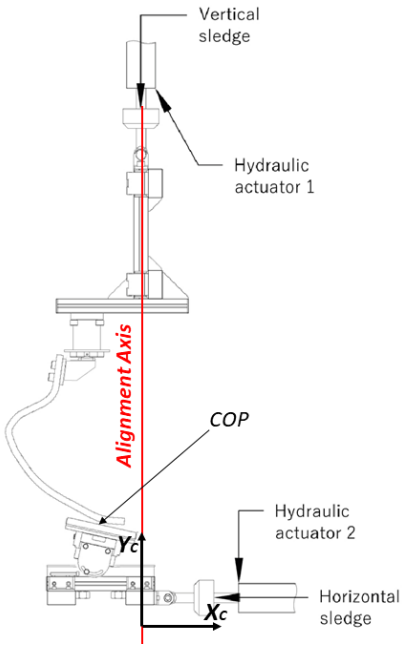


Figure 4.5. Colossus' degrees of freedom.

During the tests, the prosthesis is hooked using to the vertical cylinder with a pyramidal joint for C-shaped prosthesis or with a system made by an aluminium plate and 4 bolts for J-shaped feet.

4.2 INSTRUMENTATIONS

The Colossus machine, besides the uniaxial load cells and the potentiometer on the cylinder, is also provided of two multiaxial load cells: a triaxial one that measures reaction ground forces (FX_G, FY_G and FZ_G) and is located under the contact plate, and a 6-axes one that measures forces and moments (Fx_C, Fy_C, Fz_C, Mx_C, My_C and Mz_C) between the clamp and the vertical sledge frame. Nevertheless, because the aim of the tests on the Colossus is typically the measurement of axial forces, displacements and calculation of the centre of pressure, only the uniaxial load cells, the potentiometers and the 6-axis load cell are used.

All data were acquired during the tests by the *SoMat eDAQ-lite* acquisition unit to which all the load cells and the potentiometer, even the ones on the cylinders, were connected.

Figure 4.6. SOMAT acquisition unit.

The *SoMat eDAQ-lite* is a microprocessor-based data acquisition system designed for portable data collection in a variety of test environments. It is a sealed stand-alone data acquisition system and has the capacity to perform signal conditioning and on-board data processing. Input power for the system operates in a wide range from 10 to 55 volts DC for the ELCPU-PLUS processor or from 10 to 18 volts DC for the ELCPU processor and internal back-up batteries protect it from unplanned power losses or low voltage events.

The eDAQ-lite consists of one base processor layer and a variable number of optional add-on layers:

- ELCPU
    1. ELCPU: base processor
    2. ELCPU-PLUS: base processor with extended voltage
- ELHLS: high level analog layer
- ELBRG
    1. ELBRG-350: bridge layer for 350-Ohm
    2. ELBRG-120: bridge layer for 120-Ohm
- ELDIO

1. ELDIO-B: digital input/output layer
2. ELDIO-5HZGPS: digital input/output layer with GPS

- ELNTB: non-isolated thermocouple layer
- ELBAT: battery layer

The *SoMat eDAQ-lite* is designed to be connected to an adequate power supply: the ELCPU-PLUS processor supports nominal 12-, 24- and 42-volt vehicle battery systems and the ELCPU processor supports nominal 12-volt vehicle battery systems only.

After planned the test and installed the various gages, sensors, load cells and cables required for the test, is important to connect the transducer cables to the appropriate eDAQ-lite layer or the measure will be compromised.

The TCE software used to manage the SoMat permits to fully define the desired test and verify that the transducers and data channels are operating as expected using TCE channel display options.

Before starting the tests, it is necessary to initialize it and after that, multiple test runs during a single test session can be performed. The system collects data at the desired sampling rate (0,1 Hz-100 kHz) from transducers and other input sources and stores them in a variety of formats analyzable with InField or MATLAB or Excel software.

4.3 PROSTHESES TESTED AND INAIL ASSEMBLY INDICATIONS

Different RPFs were tested on the Colossus machine over the time in order to evaluate their strength and stiffness. Every RFP is identified by a stiffness category that is related to the body weight of the athlete.

Even the same model of foot (same shape) can be bought in different categories, because the prosthesis brands, with the same moulds, manufacture various composite layups in order to obtain different stiffnesses.

However, it is not easy to define the stiffness of a RPF in an universal way due to the fact that its values changes also in function of the way the prosthesis is assembled: all the experimental tests that have been carried out took in consideration manuals indication, together with practical technicians' experience.

The picture below shows the indications that must be taken in account during the foot mounting, both in the in-vitro and in-vivo experimental tests. INAIL technicians during in-vivo applications, mark an *Alignment Axis* with a plumb line or a laser bubble. This axis has to pass through the Hip Joint Centre, approximated with the position of the greater trochanter, and the Knee Joint Centre. When the residual limb or the RPF are in a maximum extension position, the distance between this axis and the tip of the RPF is called tap, and it is the reference parameter that is regulated in-vivo.

In the Colossus test bench, the alignment axis coincides with the axis of the vertical cylinder. Therefore, for C-shaped prosthesis, the tap can be replicated in the machine by acting on the pyramidal joint that connects the clamp to the Colossus; for J-shaped feet, both tap and socket tilt are reproduced by some aluminium spacers and a pyramidal joint.

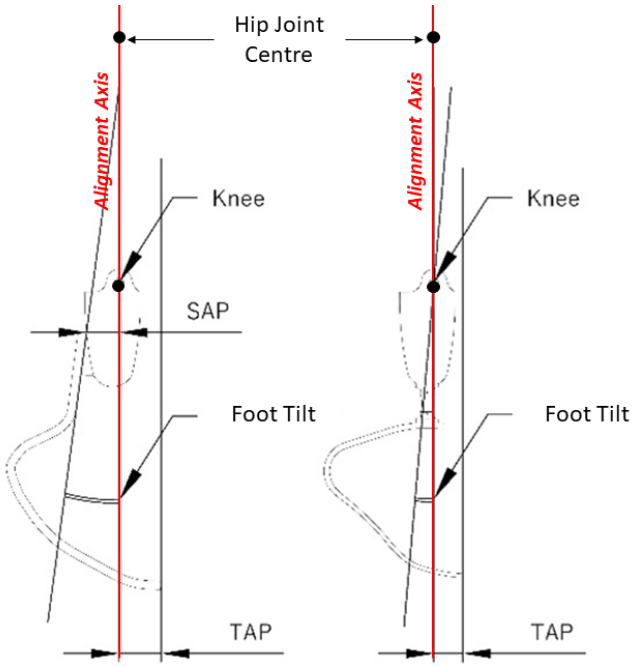


Figure 4.7. INAIL assembly indications for RPFs.

4.3.1 Reference systems

There is no standard reference frame for RSPs in literature, so it is necessary to explain the ones adopted to evaluate forces, displacements, and angles. The main ones are listed in figures 4.8 and 4.9.

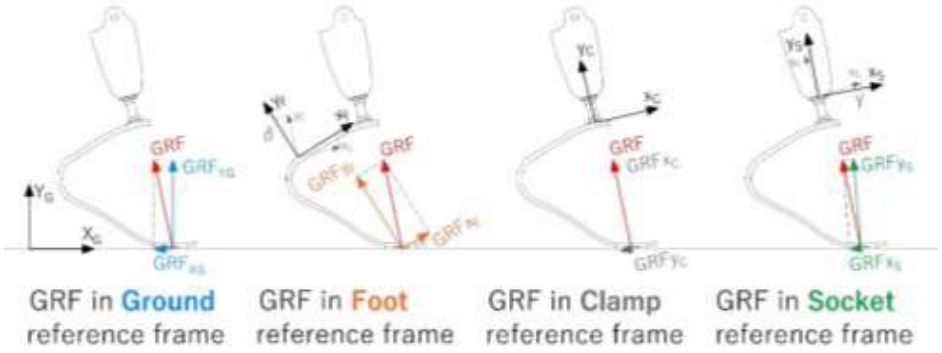


Figure 4.8. Different reference systems for C shape RPFs. (Petrone et al., 2020)

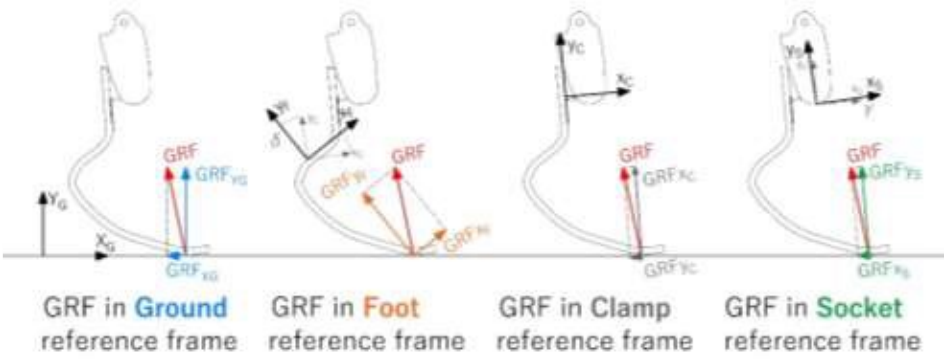


Figure 4.9. Different reference systems for J shape RPFs. (Petrone et al., 2020)

The reference frames seen in the previous figures are described as follows:

- **Ground (or Absolute) Reference System** (X_G, Y_G, Z_G): the origin of the reference frame G is located at a fixed point of the ground. The X_G axis is parallel to the ground in the sagittal plane with a posterior-to-anterior direction, the Y_G axis is orthogonal and directed upwards. The Z_G axis derives from the right-hand rule.

- **Foot Reference System** (x_F, y_F, z_F): the origin of the reference frame F is located right before the main curvature of the foot. The x_F axis is parallel to the blade, while the y_F axis is orthogonal. The Z_F axis derives from the right-hand rule.

- **Clamp Reference System** (x_C, y_C, z_C): in C-shaped prostheses, the origin of the reference system C is the geometric center of the base of the pyramidal attachment. The y_c axis is the axis of the pyramidal attachment. For J-shaped prostheses, the origin is the medial point between the screws. The y_c axis is the axis of the proximal straight stretch, while the x_c. For both types of prostheses, the x_c is orthogonal to the y_c with a posterior-to-anterior direction and the z_c axis derives from the right-hand rule.

- **Socket (or Shank) Reference System** (x_S, y_S, z_S): the origin of the reference frame S is the same as in the foot reference system. Approximating the shape of the socket to an axisymmetric solid, the Y_S axis is the axial-symmetry axis of the socket. The X_S axis is orthogonal to the Y_S axis with a posterior-to-anterior direction. The Z_S axis derives from the right-hand rule.

If the RPF is aligned correctly on the test bench the Socket Reference System has the same orientation of the Clamp Reference System and they are both inclined by an angle equal to θ_G with respect of the Ground Reference System.

The Clamp Reference System, during the in-vitro tests will have his X and Y axis coinciding with the vertical and horizontal cylinders. On the other hand, during in-vivo tests the Clamp RF will follow shank rotation with its Y-axis coinciding with the alignment one, thus it will be inclined with respect of the Ground RF by an angle equal to θ_G.

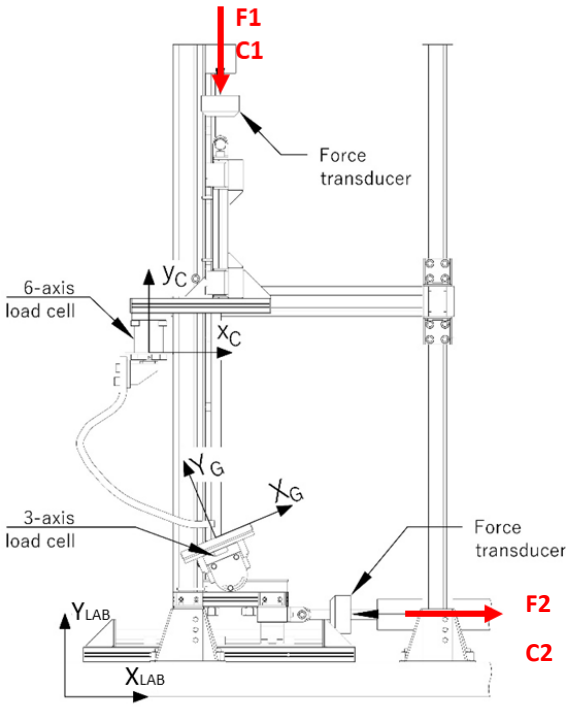


Figure 4.10. Colossus Reference System.

4.3.2 Shank angle

Another important definition is the absolute leg angle ϑ_{SH}: it is defined as the angle between the socket axis Y_S and the normal to the ground. Of course, this angle is not constant during the step. Setting the relative orientation between the Clamp reference System and the ground is the same as reproducing the relative orientation θ_G.

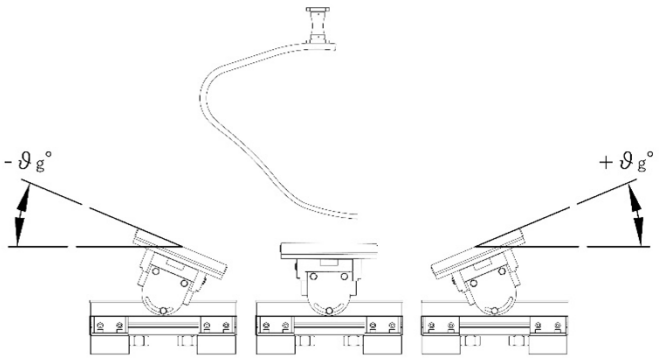


Figure 4.11. Platform inclination angle convention.

If the socket frame is considered to be fixed, the absolute leg angle can be obtained by simply inclining the slope. So, during the in-vitro tests, the ground surface is rotated to recreate different contact frames. This angle will often be called θ_G because the platform is the rotating element of the bench system. Negative values of θ_G are referred to the first braking phase of the step, while positive angles are connected to the final propulsive phase of the step.

4.4 RUNNING STEP ANALYSIS

In literature, for example in Nicola Petrone's work (Petrone et al., 2020), there are many diagrams that show the trend of the Ground Reaction Forces both in Ground and Clamp Reference Systems.

The Clamp Reference System, as described in the previous paragraph, is the one with the Y-axis coinciding with the alignment axis. The latter is swinging during the step and its angle between the vertical axis is θ_G.

Its trend can be calculated during an in-vivo test through High Frequency Video analysis or through Motion Capture acquisition.

Knowing the trend of θ_G is the only way to transform the GRFs from the Ground Reference System to the Clamp Reference System through a rotation matrix following the expression:

$$\begin{Bmatrix} Fx_C \\ Fy_C \end{Bmatrix} = \begin{bmatrix} cos\vartheta_G & -sin\vartheta_G \\ sin\vartheta_G & cos\vartheta_G \end{bmatrix} \cdot \begin{Bmatrix} GRFx \\ GRFy \end{Bmatrix}$$

The two forces F_{xC} and F_{yC} are those imposed on the Colossus to replicate the real loads for a certain angle θ_G. Once the forces in the Clamp Reference System have been calculated, a new parameter can be evaluated:

$$\rho = \rho_C = \frac{Fx_C}{Fy_C}$$

Three examples of running steps of different athletes in different conditions are reported in the following graphs: on the left the GRFs in the Ground Reference System and the trends of θ_G are presented; on the right, the GRFs in the Clamp Reference System and the ρ trends are shown.

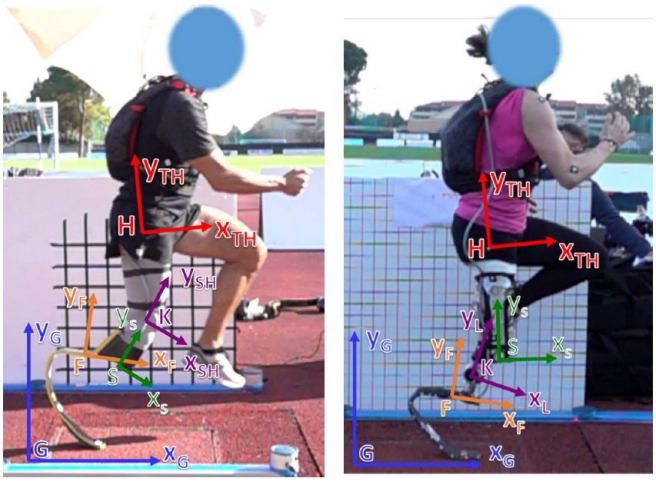


Figure 4.12. Left: Simone Manigrasso, Transtibial Amputee, J-shaped RPF. The Clamp RF in the picture is named SH. Right: Martina Caironi, Transfemoral Amputee, C-Shaped RPF. The Clamp RF in the picture is named L.

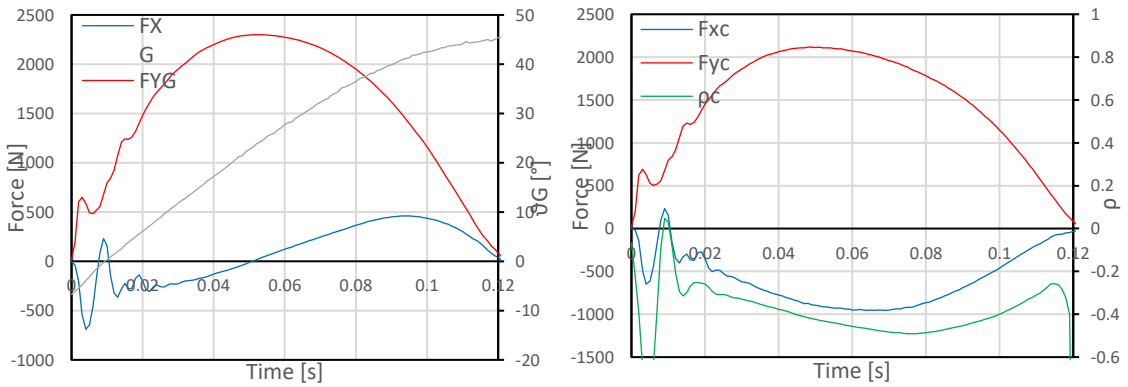


Figure 4.13. Simone Manigrasso Session4_Run2 Ground (Left) and Clamp (Right) reference systems.

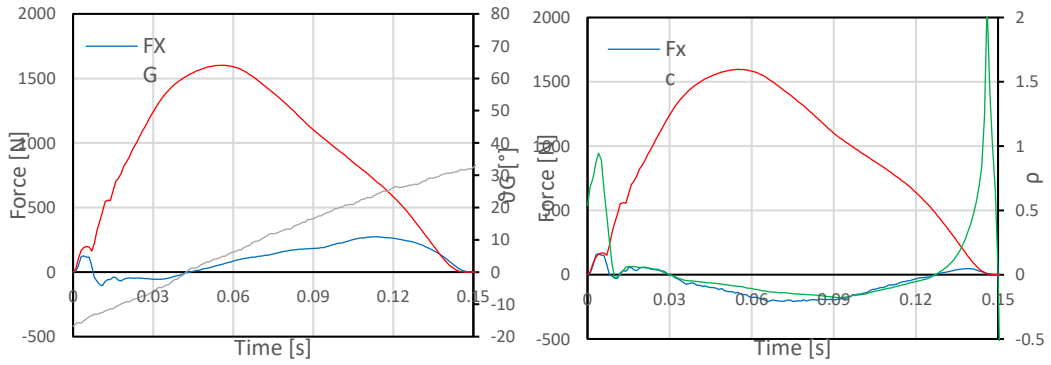


Figure 4.14. Monica Contrafatto Session3_Run2 Ground (Left) and Clamp (Right) reference systems.

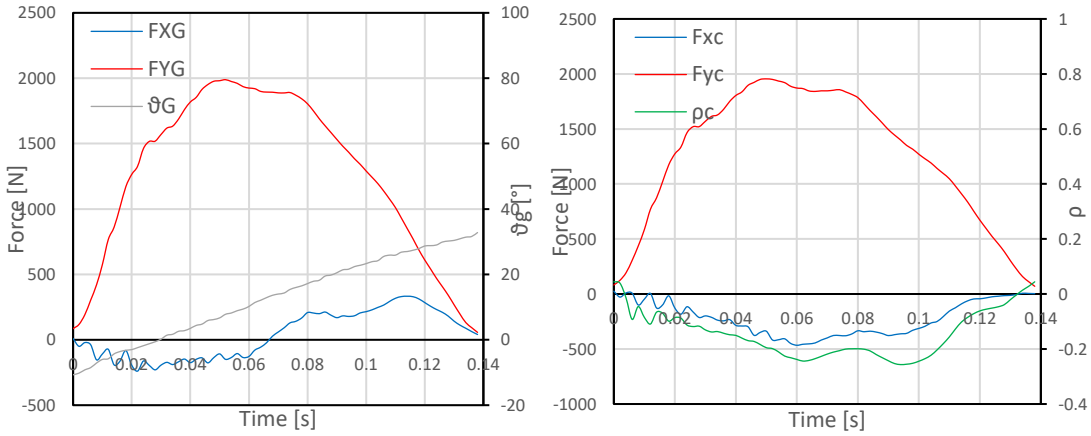


Figure 4.15. Martina Caironi Session11_Run7 Ground (Left) and Clamp (Right) reference systems.

The first step is Simone Manigrasso's (65kg) one and was done on a transtibial J-shaped Ossur Cheetah Extreme CAT 5 on the track; the second one is done by Monica Contrafatto (52kg) on a transfemoral C-shaped Ottobock Runner Standard CAT 3 on the track; the last is a step of Martina Caironi (58kg) on a transfemoral C-shaped Ottobock Runner Standard CAT 4 on a treadmill in Japan.

From the graphs it can be observed that the trends of the forces, ρ and θ_G changes with the reference systems and as a function of the % of the stance phase.

Frome the ranges of variation of these variables, some standard parameters for the tests performed on the Colossus can be established in order to create a test protocol for the RSPs that can reflect their real experimental load conditions.

Here are reported the peaks and valleys value of the 3 steps:

Athlete	F_{xGmin}	F_{xGMAX}	F_{yGMAX}	θ_{Gmin}	θ_{GMAX}	F_{xCmin}	F_{xCMAX}	F_{yCMA}	ρ_{Cmin}	ρ_{CMAX}
Manigrasso	-257	460	2298	-6,7	46,1	-954	0	2117	0,49	0
Contraffatto	-56	273	1602	-16,8	31,7	-159	100	1601	-0,13	0,11
Caironi	-299	330	1990	-10,7	32,8	-466	0	1955	-0.25	0

Table 4.1. Significant Values of Loads, θ_G and ρ from 3 In-vivo Steps.

The fluctuation of the values between different athletes, but also between different steps of the same subject, is significant, therefore it is difficult to determine a test method that could replicate all the possible load conditions to which a RPF can be subjected.

Therefore, seven different θ_G that could best fit the in-vivo variation ranges without requiring a huge number of tests on the same RPF, were initially selected as standard test parameters from the possible slope adjustments: -22.5°, -15°, -7.5°, 0°, +7.5°, +15°, +22.5°.

In a similar way 3 different ρ values were set as standard: -0.2, 0, 0.2.

Since Neural Networks require a big set of data to acquire good precision of prediction during the training phase, for the sake of this thesis more angles and values of ρ of were considered: a new flange with a continuous fissure was used to reproduce more the angles and 0.1 and -0.1 were used as values for ρ. Moreover, procedures with a variable value of $\rho(t)$ were considered for the bench tests carried out on the Colossus.

Chapter 5: In vivo running tests

This chapter's purpose is to give a detailed description of the acquisition of data from in-vivo track tests carried out at Padua's Palaindoor with elite paralympic athlete Ambra Sabatini, gold medalist at Tokyo 2020 Olympics, during the OS3 session. The Palaindoor is a building that hosts various sports equipment and tracks, among which an Olympic running ring and a 100m sprint track are present. The tests were performed using the latter.

A brief description of the tests from a previous session (OS1) with athlete Martina Caironi will also be given, since its data have been used in this work.

The kinematic data, acquired with both optical motion capture and regular cameras, will be confronted in this chapter. The kinetic data, which were acquired with a 6-axial load cell placed on the prosthesis' clamp, will be used for the validation of the Neural Network model, as seen in later chapters.

5.1 INSTRUMENTATION

The tests were carried out on an instrumented track, composed of:

- System of 25 cameras **Vicon Vero v2.2,** 2 cameras **Vicon Vantage v16**, 1 camera **Vicon Vantage v5.**
- System of 7 cameras **Vicon Vue**, used for markerless Motion Capture.
- 2 **high-speed cameras**: dynamic video acquisition in the sagittal plane from the left and steady in the frontal plane from the front.
- 1 **slow motion camera** for steady video acquisition in the sagittal plane.
- **DTS Slice Nano**; **DTS 6DX PRO-A** (Triaxial accelerometer + triaxial gyrometer), acquisition frequency 2000 Hz; 6-axis **load cell** SRI M3564F1, acquisition frequency 2000 Hz.
- Dynamometric **Starting Blocks** equipped with load cells (**Gefran CU K5D** and **CU K1C**), acquisition frequency 1000 Hz.
- Optical system for movement detection **OptoGait.**
- 3-Axis Logging Accelerometer **AX3 Axivity,** acquisition frequency 200 Hz.

Two types of tests were conducted:

1) **Track steady state running** (TSSR): TSSR consists in starting from standing, outside the acquisition volume and reaching a constant speed, equivalent to the 60% of the personal maximum speed. This speed is maintained in the acquisition volume.
2) **Start**: As regards START tests, the athlete started from the blocks, placed in the acquisition volume, and reached the 60% of the personal maximum speed.

The athlete is a transfemoral amputee (category T63). She used a single configuration of prosthetic elements for all the tests, consisting of:

- Socket S3 with flexion angle approximately 15°;
- Ottobock 3S80 prosthetic knee;
- Ottobock Runner Standard foot CAT 3.5, tap approximately 5 cm.

Figure 5.1. Prosthetic leg used during the tests.

In the following sections, the instrumentation used for the scope of this thesis will be described.

5.1.1 Vicon Nexus

The instrumentation used for the acquisition of kinematic data for these tests is the *Vicon Nexus* optical motion capture system. Optical motion capture (generally abbreviated in MOCAP) consists in the process of capturing and reconstructing body movements with the use specific hardware and software. MOCAP is used in a variety of fields, ranging from gait analysis for medical purposes to computer animation.

In particular, the Vicon Nexus system is an optoelectronic system that makes use of infrared cameras able to track the movement of body segments. The tracking often exploits the use of *markers*, which are spherical reflective objects accurately placed on the athlete's body to highlight the key anatomical landmarks. Several protocols for the placement of markers exist, such as the Davis protocol (20 markers) and the Helen Heyes protocol (15 markers). For paralympic athletes, the placement of the markers on the prosthetic elements is done depending on the type of amputation, socket, and blade.

More technically advanced systems allow for *markerless* acquisition, tracking body surfaces instead:

Figure 5.2. Vicon Vero camera.

For the purpose of these tests, Vicon cameras were placed on the sides of a whole running lane. Seven Vicon Vue cameras, used for markerless acquisition, were placed at the beginning of the acquisition volume, while the rest of the cameras were Vicon Vero and Vicon Vantage cameras used for regular markered tracking. Approximately halfway through the lane, a metal frame was placed to hold cameras at a higher level from the ground.

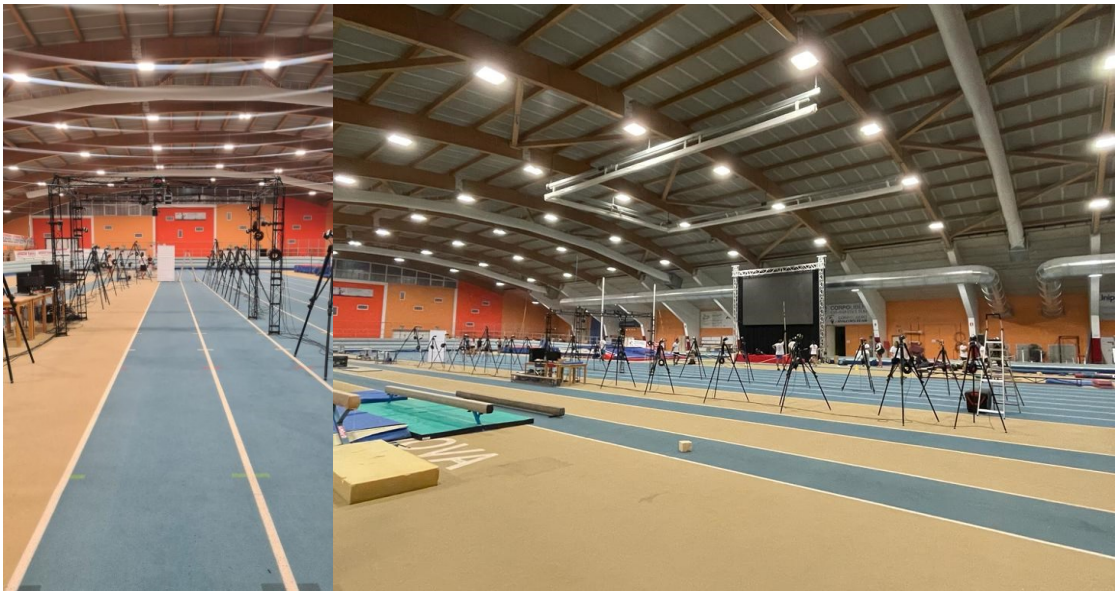


Figure 5.3. Instrumented track. The cameras were placed on several tripods to covers the full distance of the lane; in the left picture, the metal frame holding cameras above the track can be seen.

Once the cameras were placed on the tripods, pan and tilt needed to be regulated for a proper orientation. Then the system could be turned on and the optical parameters of the cameras, being aperture, zoom, and focus, were regulated on a fixed subject.

Then, the system has to be calibrated: during this procedure, cameras detect a set of axes which is the common reference system. This is done by placing a triad of perpendicular wands inside the acquisition volume, that cameras will recognize as the absolute reference system.

5.1.2 Slow-motion camera

A fixed camera with the slow-motion feature (240 fps) was placed halfway through the track. The placement of this camera was studied to capture a single gait both in TSSR tests and in START tests, with the stance phase acquired in the center of the framing to minimize distortion. The setting of the camera was done following some quality rules, apt to have an easier tracking process during the elaboration phase:

1) The camera was set up on a tripod, as stable and fixed as possible, to minimize errors from its movement.
2) The camera's plane was aligned with the plane of motion, to have the camera's optical axis orthogonal with the running lane.

3) The camera was placed at a proper distance from the running lane to the scene to minimize distortion.
4) A non-distorting lens was chosen.
5) Four pieces of red tape were placed on the edge of the running lane, two at the left and two at the right, at a known distance. They were used as the calibrating objects during the video tracking, in order to set the dimensional scale.
6) The camera's flash was kept on during every test to clearly highlight the markers. This made the markers easier to follow on screen, leading to a more accurate tracking.

5.1.3 DTS Slice Nano

The DTS Slice Nano is an ultra-small (26x31 mm), low-power, high-shock-rated data acquisition system. It consists in a standalone system with microprocessor, memory, sensor excitation and signal conditioning. Its structure is similar to SoMat's one, since both are built with operative slices with different functions.

In the track tests, this instrument was held in a running backpack while connected to the 6-axial load cell placed in the prosthesis' clamp and used to register the forces acquired by the cell. A low-voltage battery was also held in the backpack to supply power to the Slice Nano. A custom-made controller was built to comfortably start and stop the acquisition on every test.

The data were acquired with the use of the DTS SLICEware software.

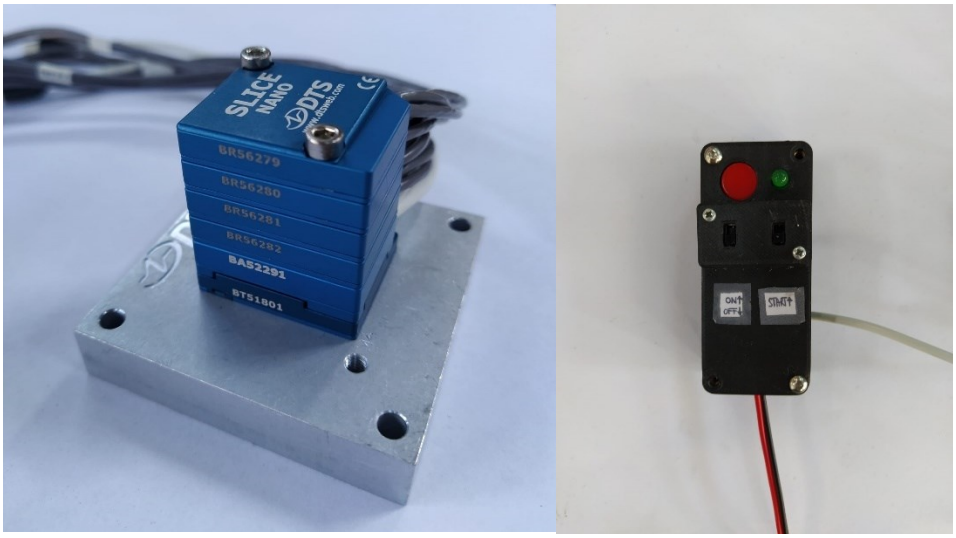


Figure 5.4. DTS Slice Nano (left) and its custom-built controller (right).

5.1.4 Load cell

The SRI M3564F1 is a thin 6-axial load cell used for the acquisition of forces and torques during these tests. Load cells are acquisition units that make use of particular configurations of strain gauges to sense forces and torques. A six axial load cell, as implied by its name, is a cell that has six channels. The linear combination of the channels gives the forces acting on the cell: the matrix containing all the coefficients needed to obtain all the forces and torques is called *calibration matrix*.

	Fx	Fy	Fz	Mx	My	Mz
CAPACITY (N/Nm)	2500.0	2500.0	5000.0	200.0	200.0	100

Table 5.1. Load cell specifics.

During the tests, the cell was placed between the prosthetic knee and the RPF in series with other connective elements. From top to bottom, the elements constituting the pylon were:

- Ottobock pyramid attack;
- Load cell;
- Steel connective plate;
- Custom holed plexiglass thickener.

Figure 5.5. Detail of prosthesis' markers and pylon from the front (left) and rear (right).

5.2 IN-VIVO TESTS

5.2.1 Preliminary operations

Once all the acquisition systems were placed in the proper position, the following operations were carried out in preparation to the test:

1) Preparation of the motion capture system. Once placed, the cameras needed to be *masked* to prevent the tracking of external objects. Then, they were calibrated with an active wand and, subsequently, with a triad to set the global reference system.
2) Placement of the anatomical markers: this process consists in the placement of markers on set anatomical landmarks. Based on their position, technical markers are placed after a static acquisition.
3) Static acquisition of the athlete to capture anatomical poses.
4) Placement of the technical markers and removal of the ones not used during the running tests. During this step, the backpack containing the DTS Slice Nano was put on by the athlete.
5) Static acquisition of the final running configuration of the markers.

The markers placed on the athlete's body can be seen in detail in figure 5.6, while the ones on the prosthesis are reported in figure 5.5.

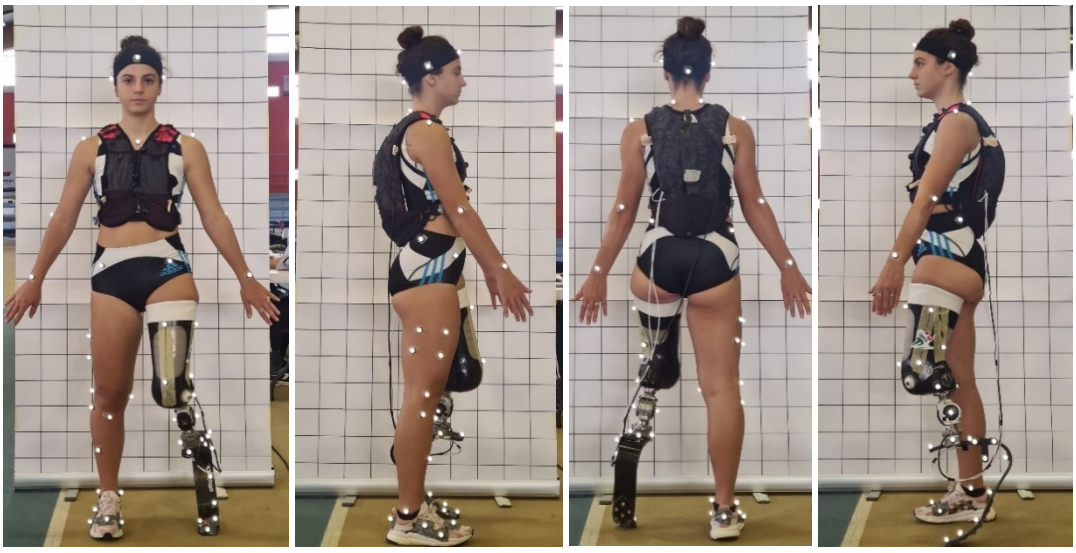


Figure 5.6. athlete AS on the sagittal and frontal plane with backpack on.

5.2.2 Testing procedure

As anticipated, two types of tests were performed by the athlete. The testing procedure for both is reported here.

1) Start of MOCAP recording procedure with Vicon Nexus.

2) Start of video acquisition with the slow-motion camera.
3) Activation of Optogait system.
4) Activation of DTS Slice Nano and start of the acquisition via the controller.
5) Start of the run. In the case of START tests, the athlete started the sprint from the instrumented running blocks positioned in the acquisition volume; for TSSR tests, the athlete started standing outside the acquisition volume and gradually accelerated until reaching a constant speed equivalent to the 60% of the personal maximum speed.

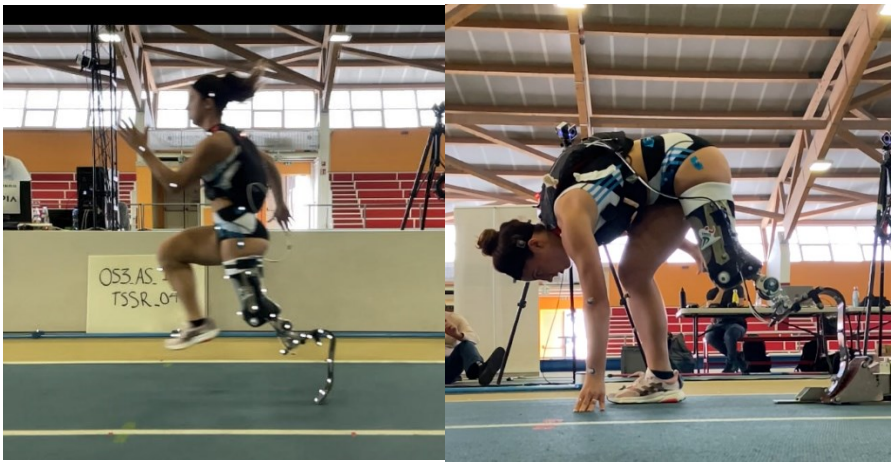


Figure 5.7. Examples of TSSR (left) and START (right) tests.

5.3 ELABORATION OF MOCAP AND VIDEO DATA

Three steps have been selected to be analyzed, one belonging to a steady state running test (TSSR04) and the other two being taken from start tests (START02 and START03). They were chosen because, for all these steps, the entirety of the stance phase was properly captured by the slow-motion camera in the center of its framing, leading to an easier and more accurate tracking process.

5.3.1 Video tracking

The tracking was done using the video-tracking software Kinovea. The video was preventively flipped with respect to the vertical axis, in order to match the orientation of the reference frame of the data obtained from the Colossus machine (as seen in later chapters). The steps followed for the tracking are now listed:

1) Upload of the flipped video.

2) Placement of a *perspective grid* with vertices coincident with the calibration objects placed on the ground. The perspective grid was then calibrated with the real distances between the tape pieces: they formed a rectangle long 100 cm on the sides coincident with the edges of the running lanes and 127 cm on the orthogonal sides.
3) Placement of a *line object* to use as the horizontal axis of the reference system. The line was placed in the middle of the running lane, where the athlete placed its RPF during the stance phase, parallel to the perspective grid. The line was then calibrated (with a length of 100 cm) and set as the horizontal axis of the linear reference frame. Kinovea automatically choses the center of the system as the first point of the line and sets the vertical axis orthogonally to the horizontal one.

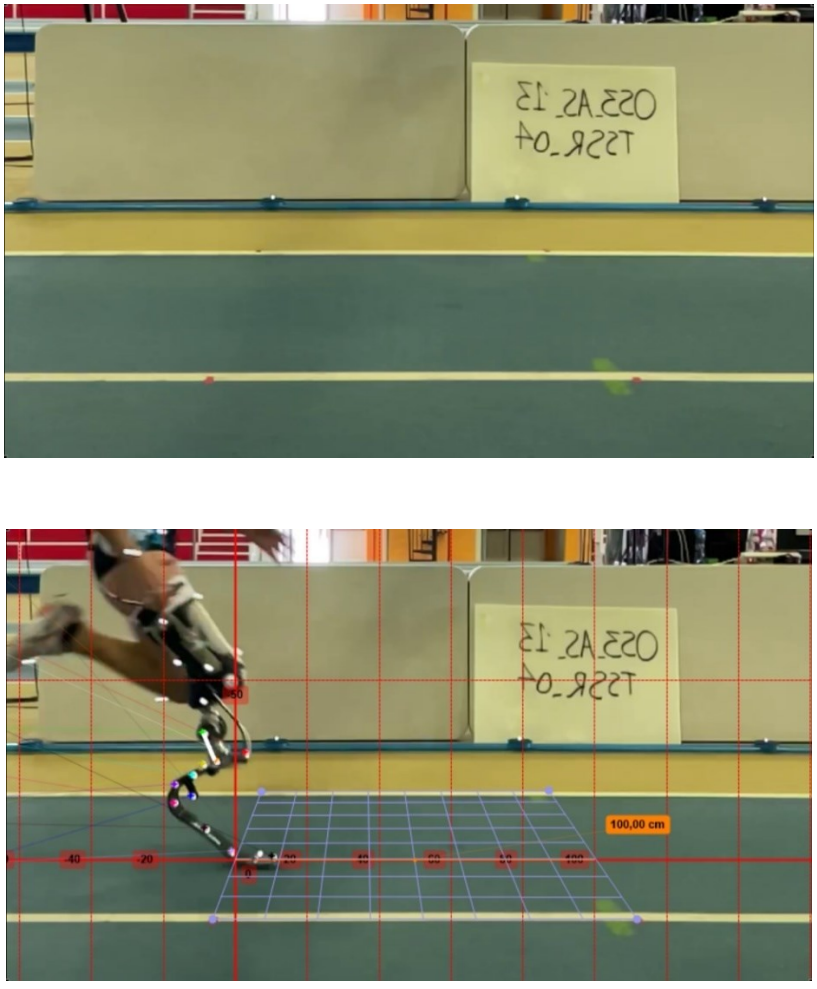


Figure 5.8. Red tape on the running lane (top); calibration grid, line and reference frame (bottom).

4) Isolation of the stance phase using the square brackets command. This command limits the

tracking to start on the frame corresponding to the initial contact and to end at the frame corresponding to the toe off.

5) Placement of visual markers on the real-life markers: in Kinovea, markers are point objects that, after enabling the "start tracking" command, track the real-life object over which they were placed. The name and color of the markers was changed to follow them easier during the tracking process.
6) Start of the tracking: after enabling the "start tracking" command, the visual markers will start to follow the real-life markers frame by frame. If the position of a visual marker is not correctly overlaid on the real-life marker, it needs to be manually adjusted in every frame. Having followed the guidelines seen in section 5.1.2, the software could properly identify the real-life markers as the objects to track and thus the visual markers did not need much frame-by-frame correction.

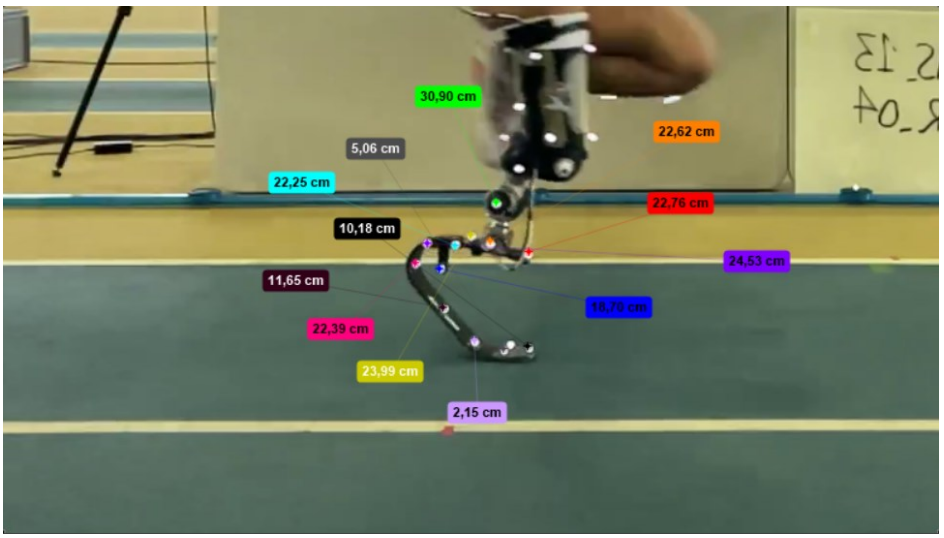


Figure 5.9. Tracking of the virtual markers.

7) Once the tracking is finished, meaning the toe off frame has been reached, the coordinates can be exported in a .csv data file using the "linear kinematics" tool. This file contain the xy coordinates of all the markers frame-by-frame, with respect to the reference system defined as in figure 5.8.

5.3.2 MOCAP tracking

The tracking of the motion capture data has been done with the Vicon Nexus software. The first step was to analyze the first frame and to create segments between markers. Each segment

needs three points to be defined in such way it is possible for the software to also identify the rotation of the segment around its axis.

When the segments needed for the analysis had been created and labelled (for instance arm, forearm, shank etc.), the markers associated with a segment could be relabeled as preferred. Markers and segments can be colored to simplify the visual interpretation.

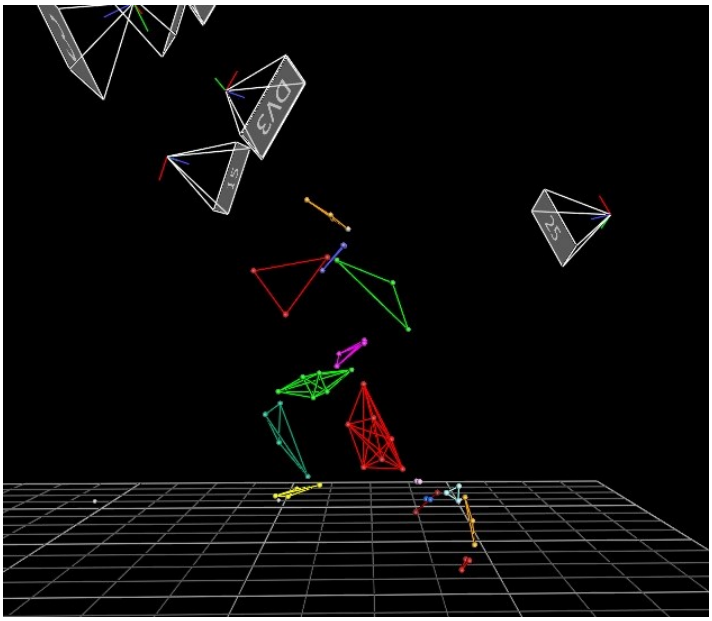


Figure 5.10. Reconstruction of body segments through the position of markers on the Vicon Nexus software.

Once all markers and segments had been identified and labelled for the first frame, the protocol could be saved in order to use the same connections path and labelling for the other tests of the same session. The protocol was be saved in a .vst file.

Then, all the other frames could be analyzed one by one. It is probable that a marker will disappear and appear again after a few frames: in that case, the software could lose its label and it has to be reassigned manually. To help identify these large gaps on the signal it is useful, to check the markers trajectory on each axis over time.

When all the large gaps were filled and each marker was correctly labelled during all the run, some smaller gaps remained. These gaps were repaired with the specific *gap filling* tool that uses various interpolating algorithm depending on the case.

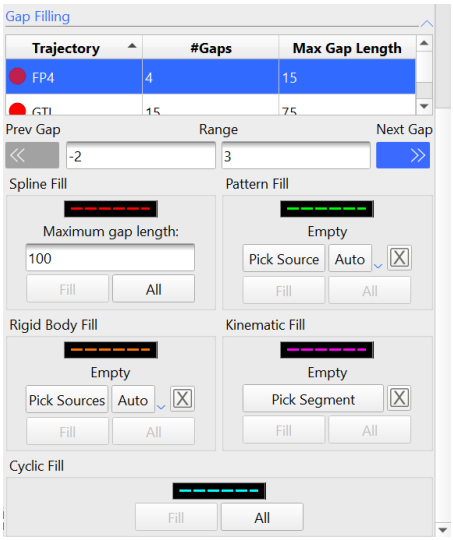


Figure 5.11. Gap filling tool.

Moreover, some spikes in the signal were generated. It was necessary to remove the problematic frames of the specific markers and reconstruct their trajectory with the gap filler tool.

Finally, *events* were created to identify the most interesting instants of the test, for instance heel strike and toe off events for both affected and unaffected leg were identified.

At the end, all the trajectory of the markers were exported in .c3d format to be analyzed in MATLAB.

5.4 MOCAP VS VIDEO

Having obtained all the data from videos and MOCAP, the two were confronted for the selected steps.

5.4.1 Data extraction

A MATLAB script was prepared following these steps:

1) Extraction from the .c3d files using the custom function `btkReadAcquisition`: this function takes a .c3d file as its input argument and stores the data in a scalar a.

```
dataTSSR04 = btkReadAcquisition('OS3_AS_13_TSSR_04.c3d');
```

2) Extraction of the markers from the scalar obtained from `btkReadAcquisition` with the custom function `btkGetMarkers`: this function takes a as its input argument and returns a vector containing two struct objects. The first one contains $m - by - 3$ matrices, corresponding to the m markers; the second one contains a scalar and a nested struct, the first being the frequency of acquisition and the second containing a sequence of strings that report the measurement units corresponding to the coordinates reported in the first struct. Since the information contained in the second struct is known, its output was suppressed by using a ~.

```
[markers1, ~] = btkGetMarkers(dataTSSR04);
```

3) Extraction and labeling of the markers in $t - by - 2$ matrices, t being the number of frames of the effective acquisition, cutting the frames to consider only the stance phase (from heel strike to toe off). These matrices have two columns because only the coordinates on the sagittal plane are taken into consideration.
4) Loading of the video data from the .csv files with the `load` function.
5) Storing of the data in labelled matrices corresponding to the markers' name.

5.4.2 Data analysis

The comparison between the two acquisition methods was carried out considering the tip-to-tip distance, being the distance from the proximal tip to the distal tip. This was done because it is the most variable measurable quantity in the foot deformation process; moreover, the tips are easy to track even from markerless acquisitions (from both videos and MOCAP), making their distance an easy variable to consider and analyze.

The analysis was computed following these steps:

1) Calculation of the tip-to-tip distance for both MOCAP and video data using Pythagoras' theorem:

$$D = \sqrt{(x_d - x_p)^2 + (y_d - y_p)^2}$$

With the subscript d and p referring respectively to the distal and proximal tip.

2) Resampling of the distance obtained via video using the `resample` function. This was done because the vectors had different lengths and thus were not comparable in every instant.
3) Smoothing of the resampled vector using the `smooth` function. The resample function uses an antialiasing lowpass filter, which can lead to discontinuity points after the resampling operation. To fix this, the variable was smoothed with a moving average filter.
4) Scaling of the distances based on the ground through of the undeformed distance. A scaling

factor was calculated as the ratio between the undeformed distance in data and the one real one measured on the RPF; then, the data were scaled accordingly.

These operations were carried out for all three steps taken into consideration, and for two other steps acquired during a previous session (OS1) done with paralympic athlete Martina Caironi. OS1 was performed entirely on an instrumented treadmill at UNIPD's sports and rehabilitation engineering laboratory, before the start of this thesis' project.

In the graphs below, the red line is the distance calculated via MOCAP while the blue line is the distance calculated via video.

OS3

OS3 is the session discussed in the previous pages. The three steps considered are TSSR04, START02 and START03.

OS3 TSSR04: this step was taken from the fourth steady state run. It is the nineth steps the athlete takes with the affected limb.

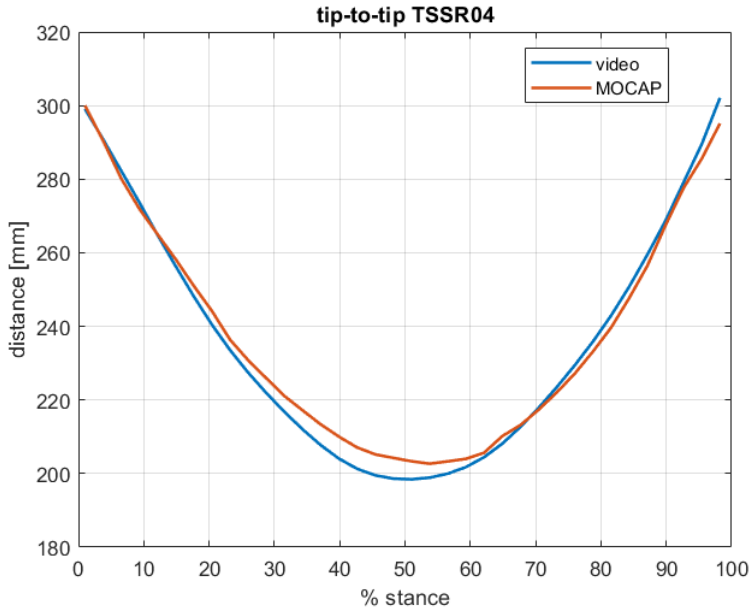


Figure 5.12. tip-to-tip distance for the selected step of TSSR04.

OS3 START02: this step was taken from the second start test and is the second step the athlete takes with the affected limb. In this step, the RPF was rotating on the longitudinal axis during the first part of the stance phase, leading to a less precise bidimensional tracking since part of

the motion takes place in the frontal plane.

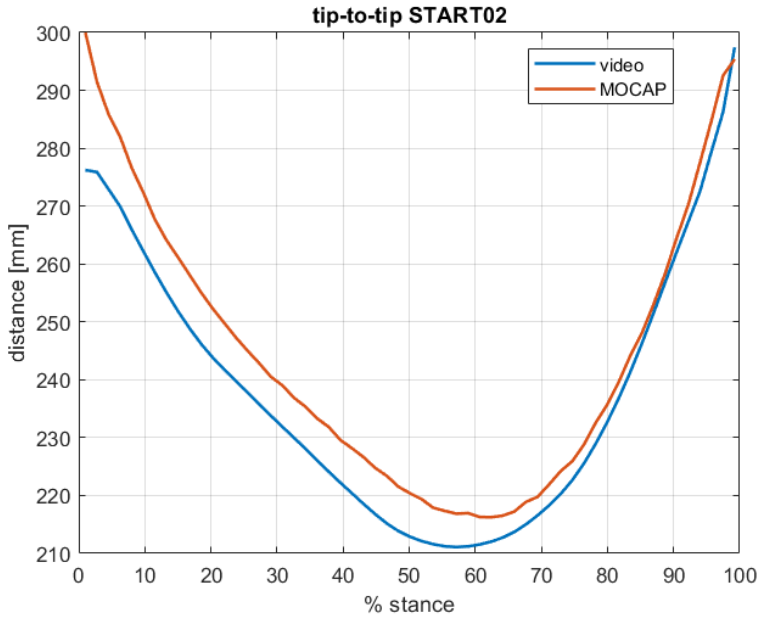


Figure 5.13. tip-to-tip distance for the selected step of START02.

OS3 START03: this step was taken from the third start test and is the second step the athlete takes with the affected limb.

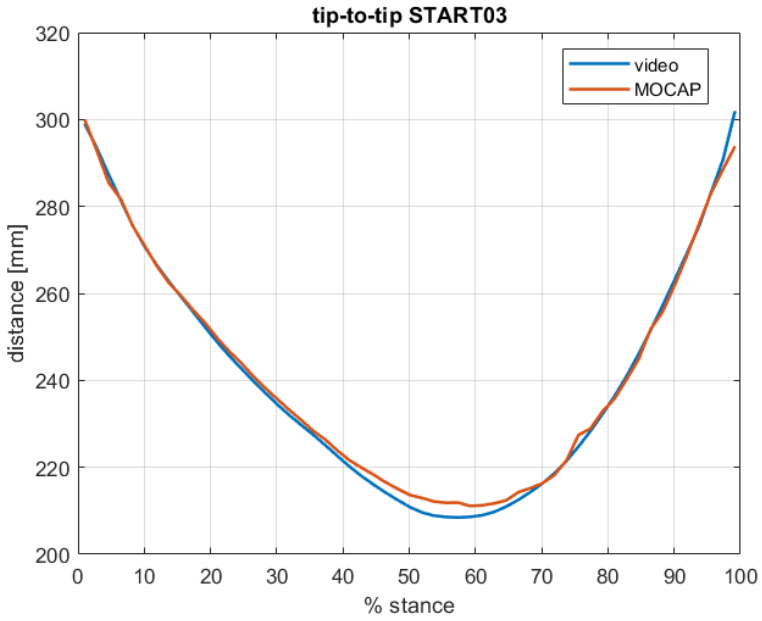


Figure 5.14. tip-to-tip distance for the selected step of START03.

OS1

The acquisition during this session were done with the following instrumentation:

- **Technogym SkillRun** Treadmill.
- **BTS Smart Capture-DX 6000** motion capture system (acquisition frequency 250Hz), with 7 cameras.
- Two **high frequency cameras** placed on the left and right of the treadmill.
- Four **P 6000 BTS** force platforms placed under the treadmill.

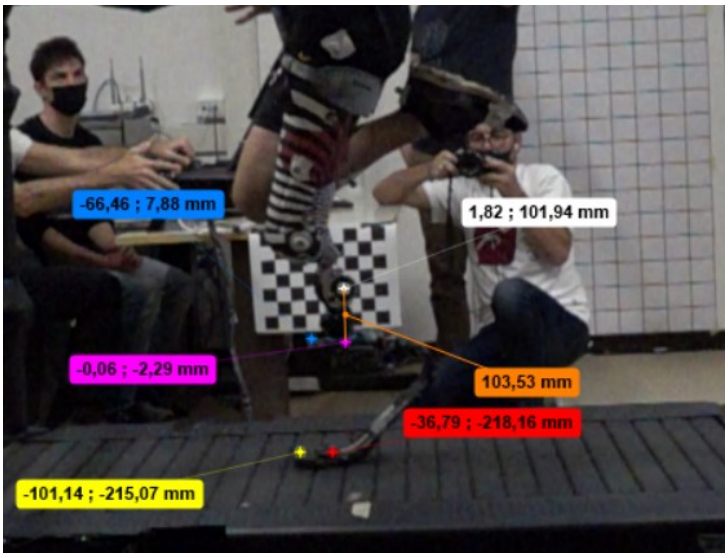


Figure 5.15. Tracking of Martina Caironi's treadmill run.

Athlete Martina Caironi is a transfemoral amputee (category T63) with the prosthetic leg on her left side. For these tests, she used an Ottobock Runner Standard CAT4 with various sockets. The forces were acquired via force platforms so the pylon didn't have a load cell as in OS3 and was a regular Ottobock pyramidal attack.

Although the placement of the markers on the RPF was done following a different protocol than in OS3, the proximal and distal tips both had markers with no significant changes from OS3.

The kinematic data acquired during this session have been elaborated during this thesis for two steps. It is important to notice that the videos captured with high frequency cameras, albeit smooth, present an insufficient illumination and thus make the image blurry and the tracking process more difficult and less precise overall. The guidelines reported in section 5.2.1 were, in fact, written to improve the tracking of data acquired with video cameras.

The steps recorded by the cameras are the last 5 taken by the athlete for every run. The steps analyzed here belong to steady state running tests, conducted at the constant speed of 16 km/h

as set on the treadmill, and are both taken from a video of the same run (SSR14) recorded from the left side.

OS1 SSR14 step 1: this step is the first step of SSR14 recorded with high-speed cameras, meaning it is the first of the last five steps.

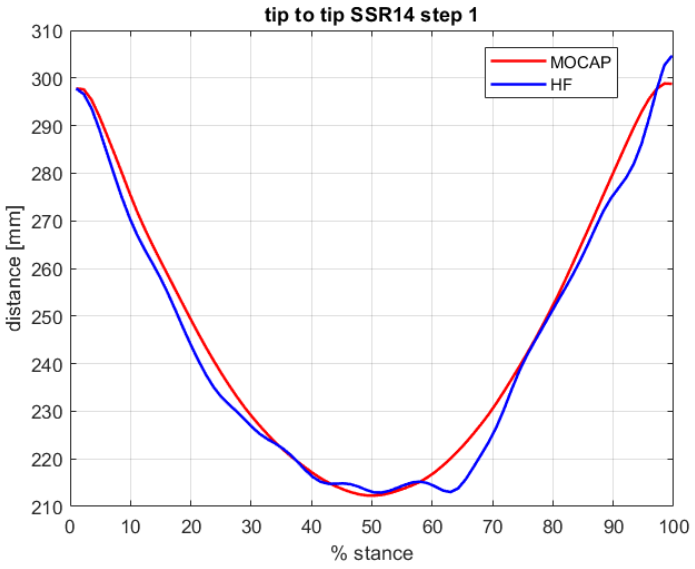


Figure 5.16. tip-to-tip distance for the first step of SSR14.

OS1 SSR14 step 2: this step is the second step of SSR14 recorded with high-speed cameras, meaning it is the second of the last five steps.

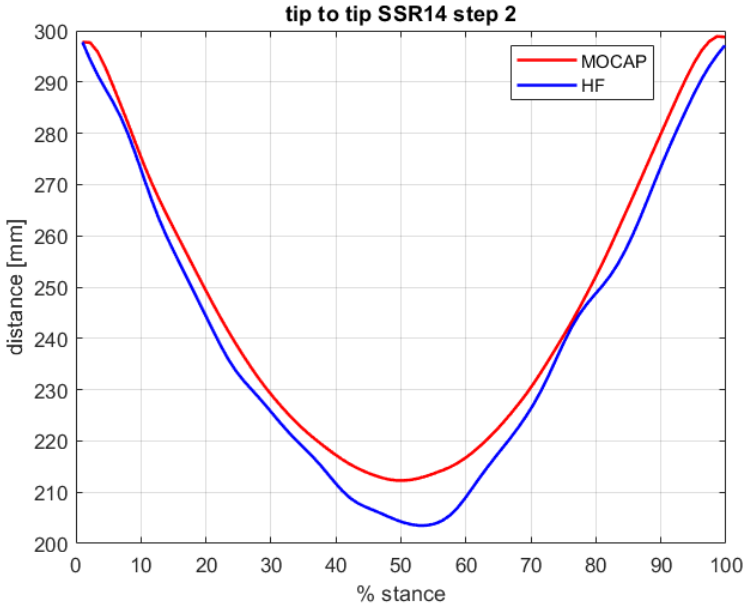


Figure 5.17. tip-to-tip distance for the second step SSR14.

5.4.3 Discussion

The percentage error between the two measures frame by frame can be computed as:

$$\%err = 100\frac{|d_{video} - d_{MOCAP}|}{d_{MOCAP}}$$

Statistical parameters referring to the percentage error for every step are reported in table 5.2.

	TSSR04	START02	START03	SSR14 s1	SSR14 s2	Mean
Mean %err	1.34	2.66	0.60	1.19	2.12	1.58
STD %err	0.84	1.30	0.53	0.93	1.03	0.92
Max %err	2.83	7.92	2.76	3.53	4.61	4.33

Table 5.2. Statistical parameters of the percentage error for every step.

As anticipated, START02 is the step with the greater discrepancy between the two measurement systems because the athlete was rotating her foot during the first half of the stance phase. This is thus to be considered and outlier among all the steps considered, and something to take into account when using video data for kinematic analysis.

Considering the mean error, there is not a high statistical difference between the five steps. Every step has an average error of less than 3% and, excluding START02 for the aforementioned reasons, a maximum error of less than 5%. It is important to notice that the steps belonging to OS1 have a maximum error higher than the ones belonging to OS3: this can be motivated by the fact that the video tracking process for OS3 was easier and more accurate thanks to proper lighting of the scene. For the same reason, the shape of the curve for OS3 steps is smoother and have less overall discontinuity.

Given these results, it is reasonable to state that kinematic data acquired with high-frequency or regular cameras give a good approximation of the data acquired with optical motion capture technology for the steps analyzed in this chapter. The fact that the data from OS1 sessions, where the setting of the camera was not optimal, did not exceed the 5% threshold on the percentage error is a good indicator of this fact.

A proper setting of the scene could lead to even more accurate results: improvements to the guidelines followed in OS3 would require the presence of a large opaque surface to place behind the scene of action, in order to enhance the contrast between the markers and the rest. Moreover, an external source of light behind the camera, like a fixed lamp, could furtherly

improve the results.

5.5 KINETICS

The kinetic data were acquired using the 6-axial load cell through the DTS Slice Nano. Specifically, the DTS acquires the six channels of the load cell individually: to obtain the kinetic quantities it is necessary to use a calibration matrix:

$$K = M \bullet c$$

K being the vector of kinetic quantities (F_x, F_y, F_z, M_x, M_y, M_z) and c being the vector of channels (c_1, …, c_6). Consequently, M is a 6-by-6 matrix of scalars. Typically, calibration matrices are found in the documentation of the load cells.

For this thesis, the quantities of interest were the horizontal and vertical forces in a clamp reference system, being the ones that the model had to predict, as well as their ratio. Said quantities have been reported in figure 5.13 for the steps analyzed of OS3 and in figure 5.14 for the ones of OS1.

Typical values of ρ during steady state running do not exceed 0.2, except for the initial contact and the toe off regions. For start tests, the athlete accelerates and thus the horizontal force assumes greater values: as can be seen in the graphs reported for START02 and START03, F_2 is mostly positive, being even higher than F_1 in some regions.

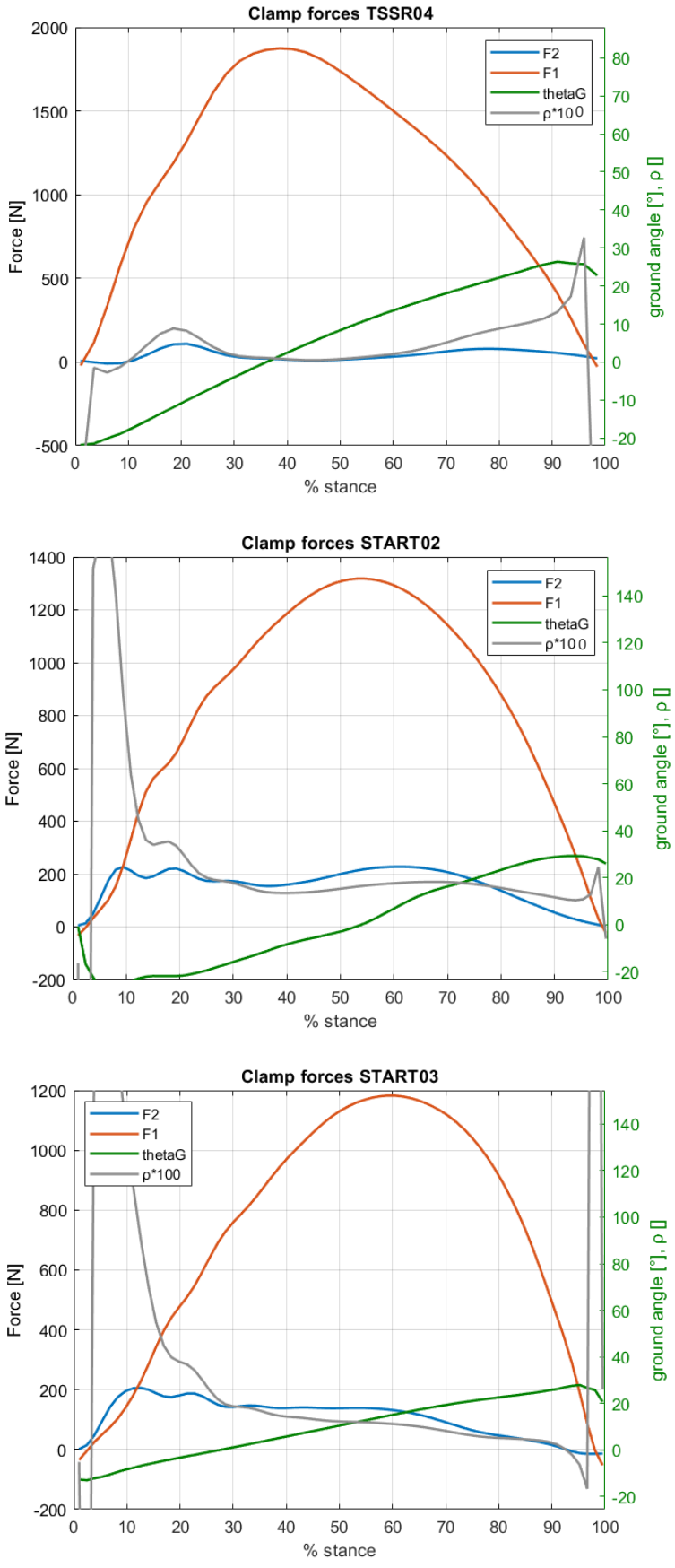


Figure 5.13. Clamp reaction forces of selected steps of OS3.

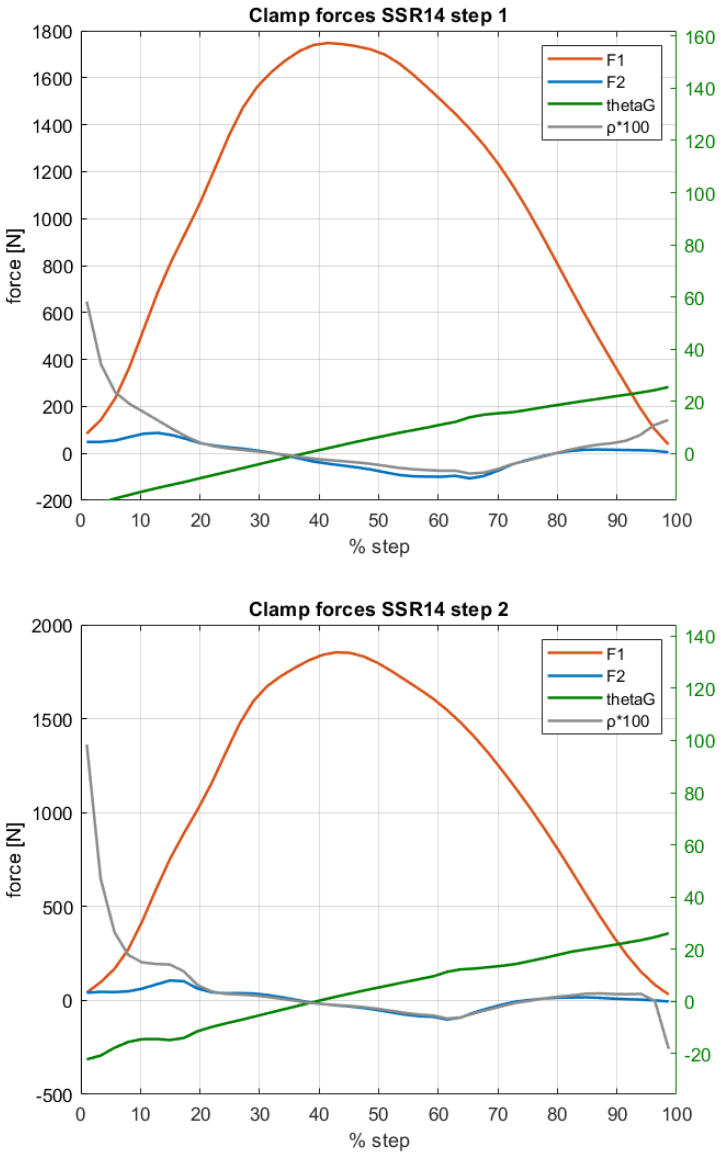


Figure 5.14. Clamp reaction forces of selected steps of OS1.

In chapters 6 and 7 two models are built and used to predict forces of the selected steps for OS1 and OS3. The kinetic data will be used as means of confrontation to understand the performance of the models on data acquired from in-vivo tests. Moreover, as will be explained in detail in chapter 7, the preventive study of kinetic data was useful to understand the range of forces corresponding to the ground angles, information that have been used to adjust the rethink the loading procedures for loading conditions that were more representative of forces acting on the foot during running.

Chapter 6: Construction of a Neural Network prototype

The first Neural Network was built and trained over data collected from bench tests carried out on Colussus machine on a Ottobock Runner Standard CAT 4. Although this network couldn't be validated in its full potential due to a mismatch between the markerset used during in-vivo tests and the one used for the training of the network, it served as a prototype on which several tests were carried out in order to optimize the parameters relative to the training process, without occurring in overfitting.
In this chapter, a detailed overview of the process is reported.

6.1 BENCH TEST PROCEDURE AND MEASURED QUANTITIES

For the solution of the problem proposed in this thesis, several tests have been carried out on the Colossus machine to acquire kinematic and dynamic data to work with. Since the tests consist in custom variations of a standard protocol to better fit the deep learning approach followed in this work, the latter is reported for ease of understanding.

6.1.1 Measured quantities

Acquisition frequency [Hz]: The maximum acquisition frequency of the load cells on the Colossus test bench equal to 100 Hz is also the one adopted. Because of the static nature of the test the data is subsampled from 100 to 1 Hz.

Tap [mm]: Distance between the clamp axis and the tip of the prosthesis. This magnitude can be reached by adjusting the Ottobock pyramidal joint.

Clamp position: for some C or CS shaped feet also the clamp position can be adjusted by sliding the pyramid joint forward or rearward thanks to a special groove at the top of the foot. This further degree of freedom permits a ±4° rotation of the foot and influences its stiffness.

Angle θ_G [°]: Slope angle from the ground of the platform connected to the horizontal cylinder. It is the first dependent variable in function of which the RPF stiffness will be defined in the next paragraphs. It is necessary to test different angles to replicate different load conditions during various instants of the contact between the foot and the ground. Typically, during in-vivo tests on transfemoral amputee subjects with a RPF, from the heel strike to the toe off, the angle between the ground and the clamp axis varies from -30° to +30°. Between these two extremes, there are seven equally spaced standard angles considered for tests. As previously anticipated, in the later tests done for this work more angles were considered in order to thicken the rose of angles for the training of the Neural Network.

Standard angles	Additional angles
0°	±11.25°
±7.5°	±18.75°
±15°	
±22.5°	

Table 3.1. Platform angles used for bench tests.

F_{xC} **AND** F_{yC} **forces** [N]: Vertical and horizontal forces read by cylinder's load cells. These are the force components in the clamp reference system that, if the RPF is aligned correctly, coincide with the socket reference system. In previous works such as Antonio Martellato's one, the standard values of the force ratio ρ were 0 and ±0.2.

Figure 6.1. Ottobock Runner Standard Cat 3 with three different clamp positions marked.

6.1.2 Standard test protocol

The standard protocol, used in previous works of thesis, makes the following assumption: referring to a running step, the F_{yMAX} corresponding approximately to midstance phase (θ_G=0) is in the range of 3 to 3.5 times body weight. Thus, this convention is adopted: for θ_G=0, F_{yCMAX} equals to the nearest multiple of 500 N to $3{,}5 \cdot BW\ [N]$. In general, the inclinations follow a conventional linear relationship:

$$F_{yMAX} = 3.5BW - 500\frac{\theta_G}{7.5}$$

The prosthesis model and stiffness category depend on the athlete's body weight. Since tests are non-destructive, the RPFs are not loaded beyond 2500N.

BW [kg]	40-50	50-60	60-70	70-80	80-90	90-100	100-120
Ossur xcel	1	2	3	4	5	6	7
Ossur xceed	1	2	3	4	5	6	7
Ossur xtreme	1	2	3	4	5	6	7
Ossur xpance	1	2	3	4	5	6	7
Ottobock runner	2	3	4	5	5	6	/
Ottobock sprinter	1	2	3	3	4	5	/

Table 6.2. Recommended body weight for all prosthesis by INAIL.

Once the maximum load to which test the prosthesis is determined, from the table just seen, the following procedure is applied for each θ_G adjustment.

1) Movement of the cylinder near limit switch to create enough space to mount the RPF.
2) Mounting of the prosthesis, controlling alignment to the axis of the vertical actuator and regulating the TAP by acting on the pyramids.
3) Placement of the clamp guide in such a way that the prosthesis, once loaded, has the sole distributed on the central part of the platform, not to generate a high M_z moments on the 6 axial load cell. This must be done taking into consideration that when the load increases, the sole will roll backwards.
4) Setting of the station limits based on the maximum force of the test. This is done for safety in the case of an error in the loading procedure: if the load exerted by the cylinders exceeds the station limits, the procedure is immediately stopped.
5) Zeroing of the forces.
6) Lowering of the vertical actuator in position for the start of the loading procedure, with a preload on the vertical actuator. To ensure the contact between the sole and the platform and to allow the horizontal actuator to properly calibrate at 0N, an arbitrary preload is given. In the tests carried out for this thesis, this value was set to 25N.
7) Start of the loading procedure on the MTS station manager.

As seen in chapter 4, the angle of the platform must be changed manually between each test.

6.2 PRELIMINARY ATTEMPTS

In the early stages of this project, some preliminary attempts to solve the problem have been carried out to understand if a deterministic approach could lead to satisfactory results. Since none of these could be used to effectively solve the problem, a deep learning approach has been ultimately developed; anyway, this chapter presents an overview of the other ideas that were initially followed and developed.

The core of the problem, approached from a deterministic point of view, consists in the identification of a number of geometric parameters (at least two) that can identify a unique geometric configuration of the prosthesis, and that allows the evaluation of the forces acting on

it. Thus, the research focus has been put on finding an ideal set which consists of two decoupled parameters, mutually independent, with one proportional to the horizontal force and the other to the vertical one.

6.2.1 Bench tests

In order to investigate further on these considerations, a set of in-vitro tests was carried out on the hydraulic bench test Colossus on an Ottobock Runner CAT 4 prosthesis. The prosthesis' notable geometrical points were highlighted with optical markers and all the tests have been filmed with a phone camera fixed in front of the Colossus machine.

More precisely, three markers were placed on the proximal zone, one on the main curvature and two on the distal end, for a total of 6 markers. The configuration is shown in figure 6.2.

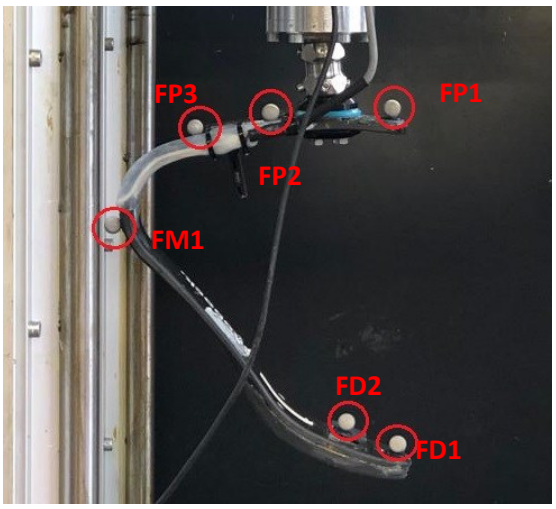


Figure 6.2. Markers placed on Ottobock Runner Standard CAT 4 for the first set of tests.

The tests were carried out following the standard protocol previously described on the seven standard angles (±22.5°, ±15°, ±7.5°, 0°). For each angle, three tests were carried out, corresponding to the three standard values of ρ (±0.2, 0), for a total of 21 tests. The acquisition frequency of the load cell was set to 100Hz.

Referring to table 6.2, the foot used for this set of tests belongs to the range of a BW of 60-70 kg. Thus, the maximum load F_{yMAX} is 2500N for 0° and decreases by 500N for every step of 7.5°.

Every test of this set was performed with the same order of operations, here reported for a maximum load of 2500N:

1) Linear load of $F_1 = 500\text{N}$, $F_2 = \rho F_1 N$ (60s)
2) Linear load of $F_1 = 2500\text{N}$, $F_2 = \rho F_1 N$ (60s)
3) Hold at maximum load for 2s
4) Linear unload of $F_1 = 100\text{N}$, F $F_2 = \rho F_1 N$ (60s)
5) The previous steps are automatically repeated for the values of ρ considered. The prosthesis is then completely discharged from the platform.

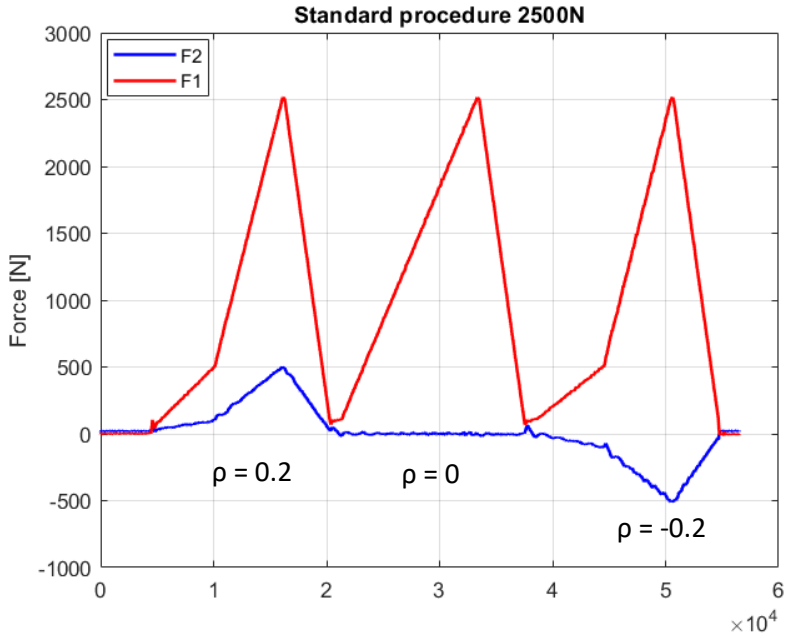


Figure 6.3. Graphic visualization of the standard procedure.

The recording phone was positioned at a distance sufficient to capture the process in action without distortion of the lens (approximately 1.5 m). Moreover, the camera was set in time-lapse mode: this is because the tests were quasi-static, with an average duration of 14 minutes to complete the three tests on a given angle. The frame rate used was 10 frames per second. Behind the camera, a lamp was placed for a better lighting of the scene, as explained in chapter 5.

The video data were extracted with Kinovea and elaborated with MATLAB.

Figure 6.4. Experimental setup. Note the two markers on Colossus' frame.

6.2.2 Approaches tested

Several approaches have been tested to solve this problem, each with its pros and cons.

Trial and error

The first method tested was trying to find the parameters with an intuitive trial and error approach, basing the choices on geometrical relationships between the parameters in order to find combinations that would eliminate the dependence to one of the forces. Based on geometrical properties, the focus was first put on quadrilateral constituted by the markers FP1-FP2-FD1-FD2, being the respectively the first two proximal and the last two distal markers in the marker set shown in figure 6.2. For example, the first attempts were done considering the relationships between the diagonals of said quadrilateral by trying to find a constant relationship with respect to one of the forces.

Several tests have been carried out, though none with satisfactory results. Still, they were fundamental for a better understanding of the problem and the relationships between the various parameters; also, they helped with the formulation and formalization of other

approaches to be tested. Some of these tests are reported below. For clearer reference, here is a legend explaining the quantities considered:

d1: distance between FP1 and FD1 (right side of the quadrilateral).

d2: distance between FP1 and FD2 (minor diagonal of the quadrilateral).

d3: distance between FP2 and FD1 (major diagonal of the quadrilateral).

dx: horizontal position of FD1 with reference to the vertical line passing in FP1.

dy: vertical displacement of FD1 from its position in the undeformed configuration.

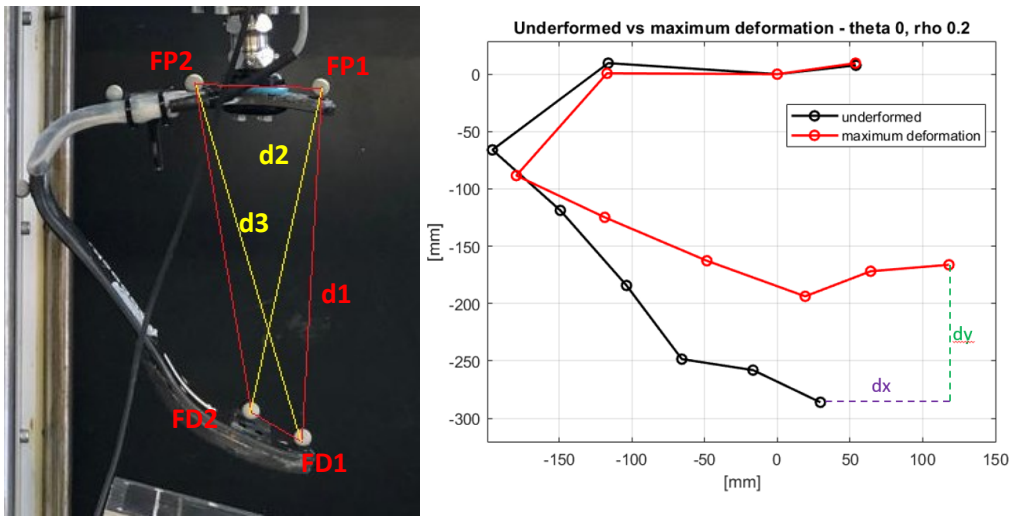
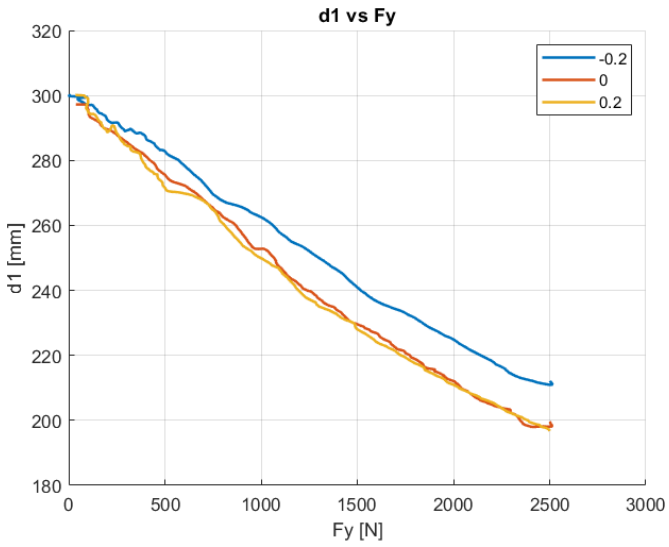
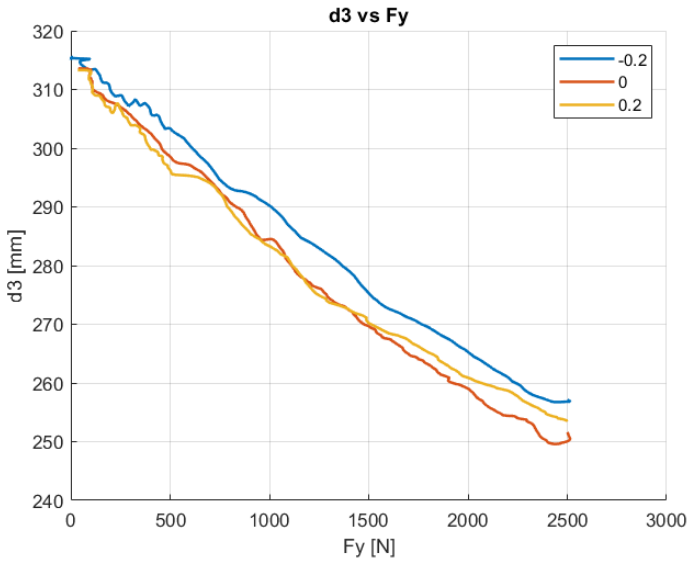
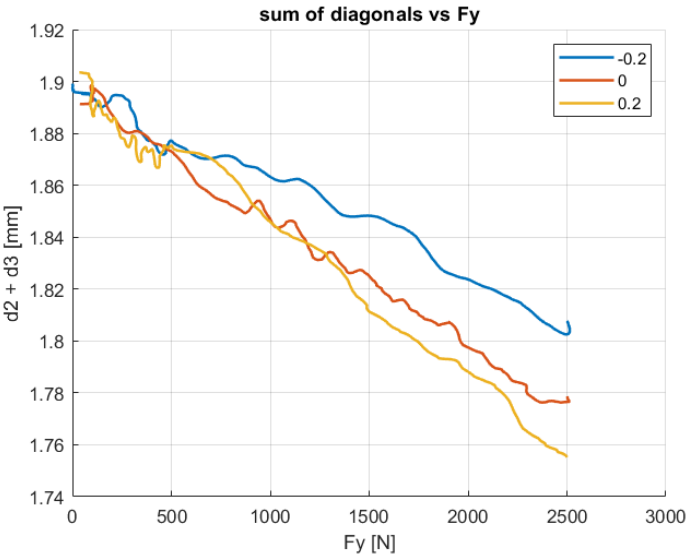


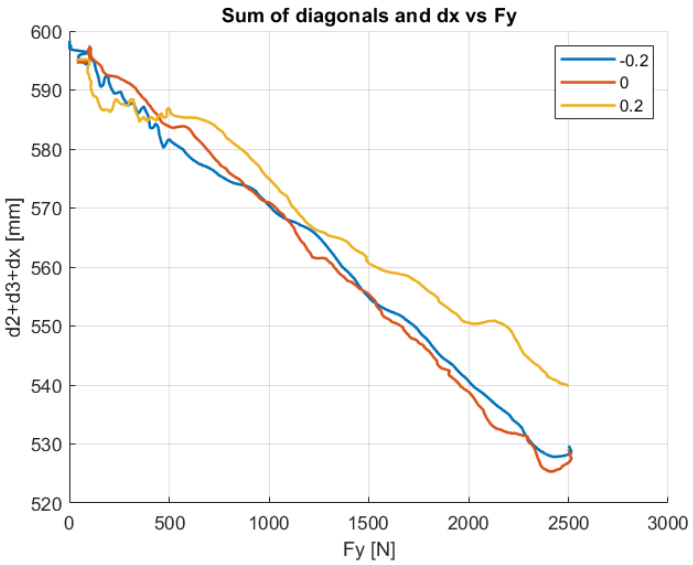
Figure 6.5. FP1-FP2-FD2-FD1 quadrilateral (left) and visual representation of dx and dy (right).

Now, a compendium of the tests is reported using the same nomenclature for shortness.




d3 vs Fy
d3 [mm]
Fy [N]
-0.2
0
0.2
320
310
300
290
280
270
260
250
240
0
500
1000
1500
2000
2500
3000


sum of diagonals vs Fy
d2 + d3 [mm]
Fy [N]
-0.2
0
0.2
1.92
1.9
1.88
1.86
1.84
1.82
1.8
1.78
1.76
1.74
0
500
1000
1500
2000
2500
3000


Sum of diagonals and dx vs Fy
d2+d3+dx [mm]
Fy [N]
-0.2
0
0.2
600
590
580
570
560
550
540
530
520
0
500
1000
1500
2000
2500
3000

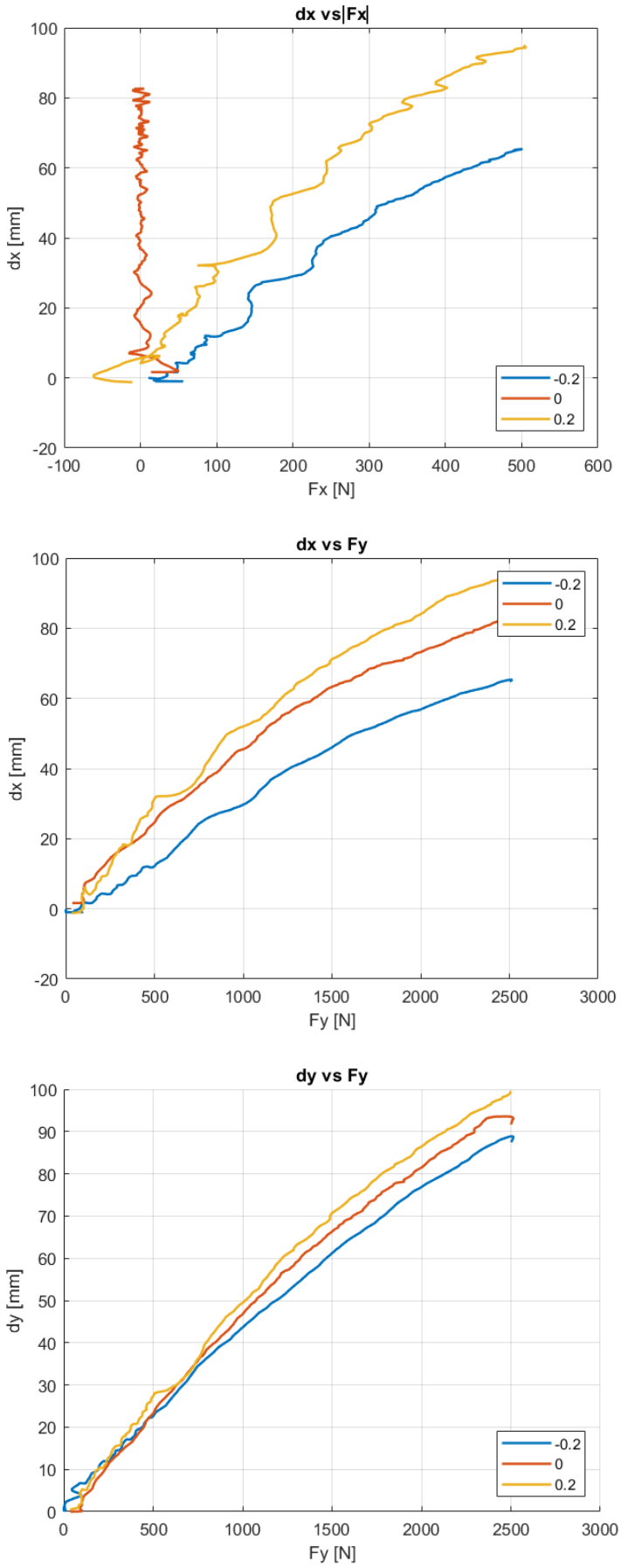


Figure 6.6. Graphs of various tests.

Calibration matrix

Having identified numerous linear parameters with the first approach, the next step was to formulate the problem with the following relationship:

$$\begin{pmatrix} p1 \\ p2 \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} F_y \\ F_x \end{pmatrix}$$

Having p1 and p2 as two linearly variable parameters and A, B, C, and D as calibration coefficients. As anticipated, the same approach was followed in the development of a FEM model by Shepherd et al.

Written this way, the problem leads back to the determination of the matrix. The equation can be explicated through the system of two equations:

$$\begin{cases} p1 = AFy + BFx \\ p2 = CFy + DFx \end{cases}$$

As previously described, the bench tests have been carried out with three different values of ρ (being -0.2, 0, and 0.2). Therefore, A and C can be determined using the results from the tests with ρ = 0 (since F_x is null); B and D can then be calculated using either of the remaining data from the bench tests. Now knowing the calibration matrix, it is possible to compute the forces from any geometrical configuration of the parameters p1 and p2.

The main advantage of this approach is its intrinsic simplicity. Though, it has some flaws:

- The bench tests do not give symmetric results. This means that plotting p1 or p2 against F_x does not give symmetrical results when considering positive and negative values, since the slope of the two functions is not equal in modulus. The direct consequence is that two calibration matrices are needed, since the values of B and D may vary from the ρ = 0.2 case to the ρ = -0.2 one.
- The calculation of Fx and Fy only works if the value of ρ is 0 or ±0.2. Since the matrix has been calibrated with values from these specific bench tests, and knowing that the results are not symmetrical for ρ = ±0.2, different values of ρ might lead to wrong results of the forces. The force ratio varies significantly during a gait *in vivo*, and is rarely stationary on ±0.2.

Statistical analysis: PCA and ICA

Another approach that has been tested is the reduction of the complexity of the problem through statistical analysis. Both PCA and ICA have been tested, with different results.

The focus behind PCA was obtain orthogonal parameters that carried good statistical significance. Carrying out PCA on 4 variables resulted in the first score having 99% of statistical significance and the remaining 3 splitting the remaining 1%. This is due to the fact that the parameters analyzed all share a similar behavior during the deformation process of the

prosthesis. Anyhow, PCA analysis could be useful in the building of a force-geometry database (approach n.4), as it can significantly reduce the dimensionality of the problem.

ICA was carried out on the same 4 variables, with two independent parameters as output. This analysis gave mixed results:

- the algorithm correctly predicted the behavior of the "source variables" (being the displacement generated independently by the forces) for the cases of $\rho = \pm 0.2$, but not for the remaining case of $\rho = 0$. This is most likely because the algorithm was programmed to give out two independent parameters, and thus it predicted both to be non-null for all the cases.
- The predicted independent variables may vary when performing two subsequent analyses, giving rise to inconsistency on repetition. Though the qualitative behavior is almost always correct, this makes it impossible to predict appropriate scaling and shifting factors to match the forces.

Database

Probably the simplest of the approaches proposed, this method does not identify two variables from which it is possible to compute the forces. Instead, it relies on the building of a database of variables that identify a unique geometrical configuration. Having tested the prosthesis, the forces acting on a given configuration are known; by coupling them with the geometric parameters, the database is built. Though, in order to build an accurate and complete database, it is mandatory to discriminate every configuration. This requires extensive testing on the prosthesis. Moreover, the database is only reliable for configurations that have been tested. This approach can be combined with a PCA analysis to reduce the dimensionality of the problem, so that less parameters need to be associated with each force combination.

Since none of these approaches gave satisfactory results, a deep learning approach was deemed to be more optimal. A new set of tests was carried out on the same RPF in order to acquire data for the training of a Neural Network model.

6.3 ACQUISITION OF THE TRAINING DATA

The training data were collected from several bench tests on the Colossus machine: forces were acquired through a load cell placed right over the foot, while the position of the markers was acquired via MOCAP.

6.3.1 Markerset

Markers were applied to the prosthesis following a custom protocol studied to maximize the visualization of the regions of interest without using too many markers. Nine markers were chosen, as can be seen in figure 6.3.

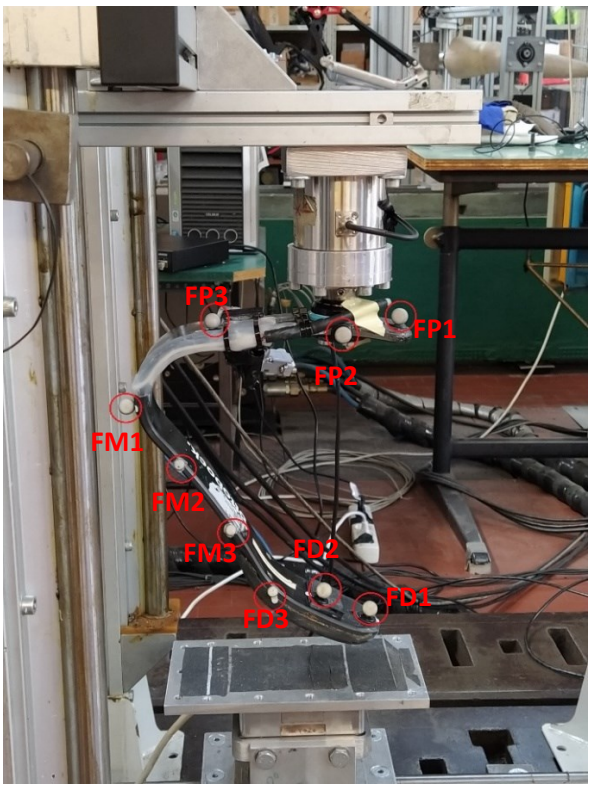


Figure 6.7. Markers on Ottobock Runner Standard for the second set of tests.

The position of the markers was chosen to highlight specific points, summarized here (top to bottom):

- FP1: proximal tip of the foot
- FP2: clamp base
- FP3: edge of the undeformed zone, right before the prosthesis' curvature
- FM1: foot curvature

- FM2: foot curvature, 75 mm from FM1
- FM3: foot curvature, 75 mm from FM2
- FD3: end of the curvature and beginning of the distal region, 75 mm from FM3
- FD2: center of the distal region
- FD1: distal tip of the foot

The positioning of the markers was chosen taking into account that the prosthesis used is an instrumented foot, so the surface was not fully available.

6.3.2 Acquisition of the data

The kinematic data were acquired with the Vicon Nexus motion capture system, using five Vicon Bonita cameras positioned around the Colossus machine on four tripods. The disposition of the cameras was studied to capture the full motion of the foot during the tests, having at least two cameras follow every marker. To fulfill this requirement, no camera was placed behind the plane of action, as can be seen in figure 6.4: four cameras were placed on the sagittal plane and one on the frontal plane.

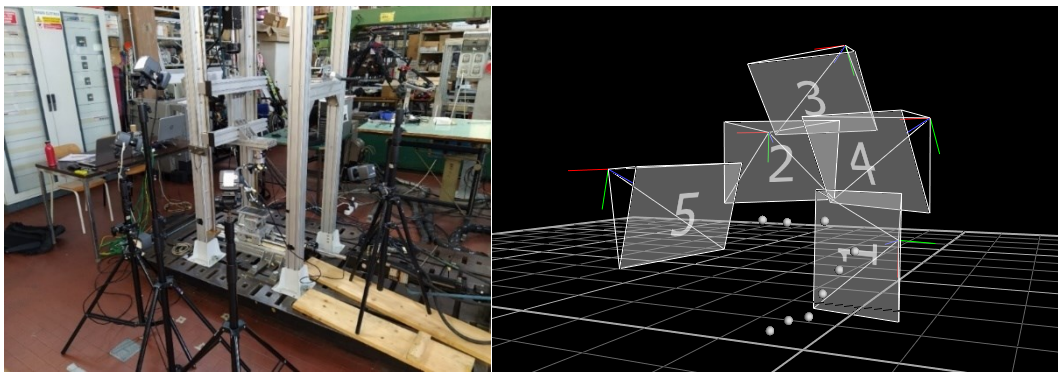


Figure 6.8. Positioning of the optical cameras around the bench (top) and their corresponding configuration in the Vicon Nexus software (bottom).

After the placement of the cameras, the acquisition was prepared with the following procedure:

1) Installation of the RPF with markers on.
2) Tuning of the optical parameters (aperture, zoom, focus) using the foot as reference. This is done by regulating three screw tuners placed on top of the cameras.
3) Removal of the RPF from the aluminum frame and masking of the cameras, to prevent the

tracking of undesired reflective objects that cameras might capture. In Vicon Nexus, the masking process is activated using the option “mask cameras” and then stopping when the volume has been entirely masked.

4) Calibration of the acquisition volume using a passive wand. The wand is a T shaped rod with three markers placed on the distal extremities. In the software, the calibration is done by using the “calibration” option with the “auto-stop” checkmark. After enabling this process, the wand has to be waved in the acquisition volume until the software stops the calibration.

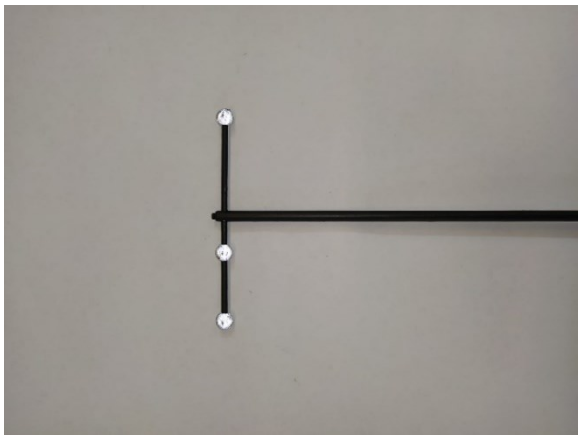


Figure 6.9. Calibration wand.

5) Setting of the global reference frame using the XY physical frame. The frame was placed on the aluminum platform where the sole of the foot would contact during the tests. The X axis was aligned with a laser spirit level. After the placement of the physical frame, the software sets it as the global reference system by using the “set reference frame” option.

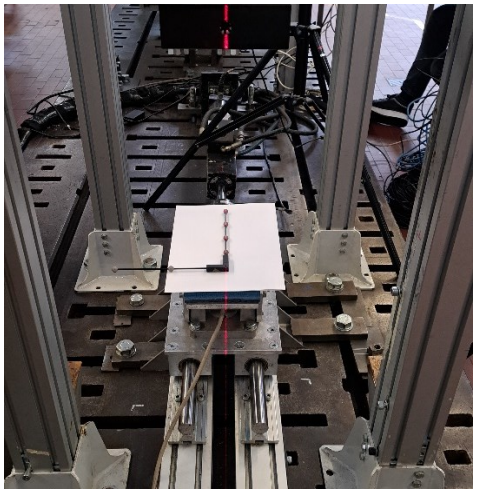


Figure 6.10 Alignment of the reference frame with laser spirit level.

6) Removal of the physical frame and installation of the foot with alignment from the frontal and sagittal plane. The alignment on the sagittal plane was done by setting a tap of 4 cm. As in the previous step, a laser spirit level was used to have a precise alignment.
7) Creation of the database where the kinematic data will be saved.
8) Acquisition of a short calibration recording. This first acquisition, typically of 300-500 ms, is done to create the virtual segments and markers to assign on subsequent recordings done on the same database.

This preparation procedure was deemed to be standard and replicated for the tests described in chapter 7, adjusting the parameters to the appropriate values.

The acquisition frequency of both the MOCAP system and the load cell was set to 100Hz: this was done in order to have synchrony between the kinematic and the dynamic data, to minimize the need of sampling in the preprocessing phase of training.

The loads were applied to the prosthesis following the standard procedure with the addition of two values of ρ, being ±0.1. This was done to have a wider range of ρ, since generally Neural Networks work better with more varied data. Due to this choice, the procedures were approximately 14 minutes long (instead of the average 9 minutes only using the standard values of ρ).

After the loading procedure was finished, the acquisitions were stopped. In the Vicon Nexus software, the data need to be processed using the core processor ("run core processor" option). This process may take several minutes, depending on the sampling rate and number of markers. After this operation, the markers were labelled using the set defined during the calibration acquisition.

The MOCAP data were saved as .c3d files, while the dynamic data obtained from the SoMat were saved in .mat format.

In figure 6.11 are reported, for visualization purposes, the plots of the markers in undeformed configuration and in the moment of maximum load.

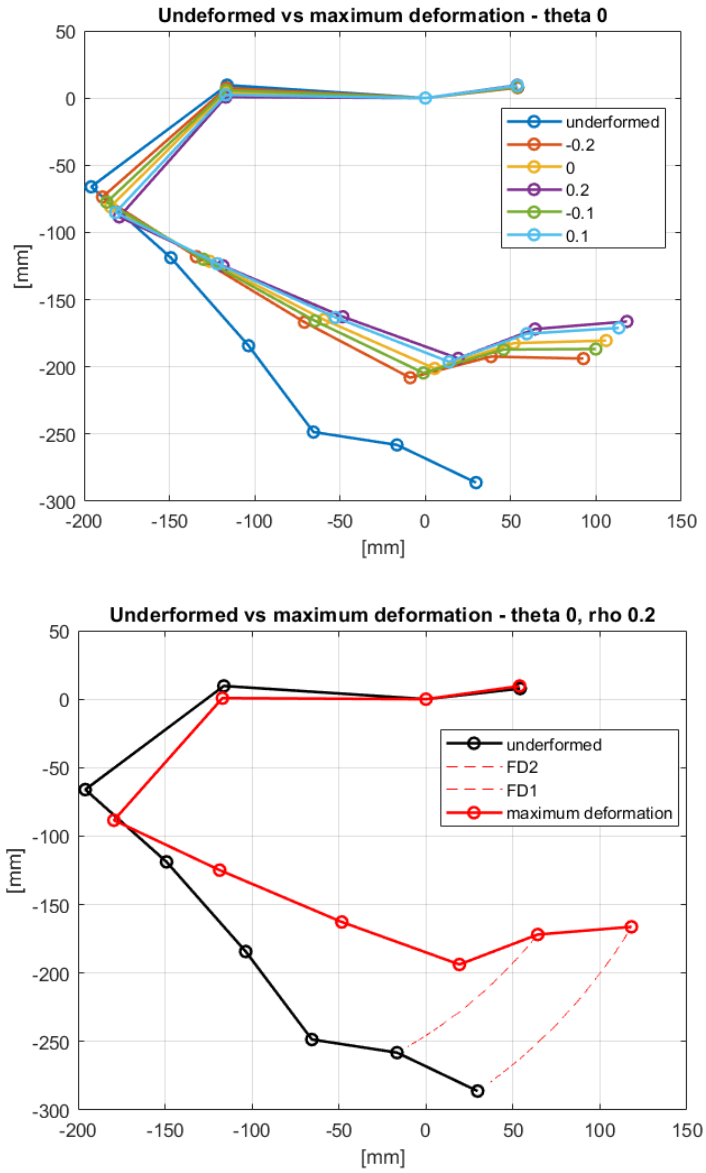


Figure 6.11. Plotting of the markers at maximum deformation for various values of ρ (top) and detail for ρ = 0.2 (bottom) at $\theta_G = 0$.

6.4 COSTRUCTION OF THE NEURAL NETWORK

The Neural Network model was built on MATLAB with the use of the Deep Learning Toolbox, the Statistical Toolbox, and the Signal Processing Toolbox.

6.4.1 Extraction of the data

The data were extracted in MATLAB following these steps:

1) Loading of the data using the `load` function.
2) Extraction from the .c3d files using the custom function `btkReadAcquisition`: this function takes a .c3d file as its input argument and stores the data in a scalar a.

```
a = btkReadAcquisition('theta_0.c3d');
```

3) Extraction of the markers from the scalar obtained from `btkReadAcquisition` with the custom function `btkGetMarkers`: as explained in the previous chapter, this function takes a as its input argument and returns a vector containing two struct objects. The first one contains $m - by - 3$ matrices, corresponding to the m markers; the second one contains a scalar and a nested struct, the first being the frequency of acquisition and the second containing a sequence of strings that report the measurement units corresponding to the coordinates reported in the first struct. Since the information contained in the second struct is known, its output was suppressed by using a ~.

```
[markers_0, ~] = btkGetMarkers(a);
```

4) Extraction and labeling of the markers in $t - by - 2$ matrices, t being the number of frames of the effective acquisition, cutting the frames before the beginning and after the end of the loading procedure. These matrices have two columns because only the coordinates on the sagittal plane are taken into consideration.
5) Storing of the markers matrices in a single matrix labelled as the corresponding angle.
6) Loading of the forces from the .mat file using the `load` function.
7) Extraction of the arrays corresponding to the forces of the two actuators.
8) Storing of the force arrays in a single matrix labelled as the corresponding test angle.

After repeating this procedure for every test set, the data were ready to be preprocessed for the training of the network.

6.4.2 Preprocessing

The preprocessing of the data consists in the operations needed to have tabular data optimized for training, testing and validation. The preliminary operation, done before creating sets, was to perform the *standardization* on both the markers' and forces' matrices, resulting in a dataset

with null mean and standard deviation equal to one (making the assumption that the data fit a Gaussian distribution). This is done using the formula:

$$x_s = \frac{x - \mu}{\sigma}$$

Having μ and σ respectively as the mean and standard deviation of the dataset x. The implementation in MATLAB makes use of the `normalize` function.

Standardization of the input datasets is usual in ML because it makes data more compatible with the structure of the model, leading to a more stabile training process since the model won't have to consider the statistical features of the data. Moreover, in the case of Neural Networks, activation functions typically have threshold points in the [-1, 1] interval, and having very distant data from those values will lead to an ineffective training. Standardization of the output is useful because the network won't have to learn the statistical features of the values used for confrontation, making the training faster and more stable and preventing the MSE to assume inconsistent values.

The coordinates of the markers were also changed to be in the clamp reference system, using the marker FP2 as its center.

The second step for the preparation of the data was to concatenate a column array of normalized angles to the markers' matrix, corresponding to the θ_g angle of the test. *Normalization*, similarly to standardization, is a scaling operation used to make the data more fit for training. It was applied setting the boundaries of the angles to be in the [-1, 1] interval: this way, the maximum value (being 22.5°) was scaled to 1 and all the other angles were scaled accordingly. The normalization formula in the interval [-1, 1] is:

$$x_n = 2\frac{x - x_{max}}{x_{max} - x_{min}} - 1$$

After these operations, the data were split in training and testing sets applying a 80/20 percentage split. Two methods were applied:

1) Method 1: the data were divided in three sets applying an 80/10/10 percentage split to the matrix containing all the observations. The first set, containing 80% of the observations, was used as the training set while the remaining observations constituted the testing and validation sets. The observations for every set were picked randomly using the `datasample` function.

2) Method 2: In this case, the testing set was constituted by the observations corresponding to one angle, while the training and validation set were extracted with a 90/10 percentage split of the remaining observations. This was repeated for every angle excluding the boundary ones, being ±22.5°. This operation, apt to understand the generalization properties of the model and prevent overfitting, is called *k-fold cross validation*, k being the number of validation sets. In general, k-fold cross validation is performed dividing the dataset in k subsets and using k-1 sets as the training set and the remaining set, called *holdout set*, for testing. Since the observations in the dataset considered are already classified by angle, the latter was used as the discriminating factor.

The application of these two methods was done to understand how overfitting could occur and affect the model in its predictions. Moreover, since the final scope of the model is to be tested on in vivo tests where the angle is a continuous function, it was desirable to understand how well the model can perform on angles it was not trained over.

6.4.3 Neural Network architecture

The network was built as a simple Multilayer Perceptron with 19 inputs, being two coordinates per 9 markers plus one angle, and 2 outputs, being the forces. The network has one input layer, three hidden layers and one output layer. The activation function for every layer is the rectified linear unit (ReLU). Every layer has 128 neurons, following the customary rule of having the number of neurons equal to a power of two (in this case, 2^7).

The code used is the following:

```
neurons = 128;
x = 19; % input variables
y = 2; %output variables
layers = [
    featureInputLayer(x, "Name", "Input")
    fullyConnectedLayer(neurons, "Name", "FC1")
    reluLayer("Name", "act1")
    fullyConnectedLayer(neurons, "Name", "FC2")
    reluLayer("Name", "act2")
    fullyConnectedLayer(neurons, "Name", "FC3")
    reluLayer("Name", "act3")
    fullyConnectedLayer(y, "Name", "FC4")
    regressionLayer("Name", "reg")
    ];
```

This network uses the syntax of the Deep Learning Toolbox. For ease of understanding, here is reported an explanation of every layer (see chapter 3 for an in depth explanation):

featureInputLayer: the input layer, takes the number of input variables as its argument.

fullyConnectedLayer: a hidden layer constituted by the neurons. Takes the number of neurons

as its argument.

reluLayer: defines the activation function to be ReLU for the previous fullyConnectedLayer.

regressionLayer: sets the output as the solution of a regression problem. This is also the layer where the loss function is defined: if no function is specified, the loss is automatically set to be the MSE.

6.4.4 Training

Several training tests were performed to understand the optimal parameters. The quality of the trained network was evaluated based on the percentage error between in the prediction and the ground truth during the testing:

$$\%err = 100\frac{|F_{pred} - F_{measured}|}{|F_{measured}|}$$

The configuration of parameters chosen, able to achieve the lowest error, is the following:

Optimizer: adam.

Initial learn rate: it is the learning rate at the start of the training process. It was set to 0.005.

Maximum epochs: the maximum number of epochs that the training can undergo if not stopped before. It was set to 50.

Size of mini batch: 256

Learn rate drop factor: it was chosen to adopt a decaying learn rate strategy, to overcome local minima in the early stages of the training and to stabilize at a lowest loss value when the training process is ending. The learn rate drop factor is the number by which the current learning rate is multiplied at a given epoch. It was set to 0.5.

Learn rate drop period: it is the number of epochs after which the learning rate drops by the value of the learn rate drop factor. It was set to 5, meaning the learning rate drops every 5 epochs.

Validation patience: this parameter regulates the number of epochs without significant changes in the validation loss that the training process undergoes before stopping early, as described in chapter 3.4. It was set to 15, meaning that if the loss computed over the validation set remains stale or gets worse for 15 iterations, the training stops.

```
%training
options = trainingOptions("adam", "MaxEpochs", 50,  'InitialLearnRate',0.005,...
    'Verbose',true, 'MiniBatchSize', 256, "Shuffle", "once",...
     'LearnRateSchedule', 'piecewise', 'LearnRateDropPeriod', 5,...
     'LearnRateDropFactor',0.5, 'L2Regularization', 0, ...
     'ValidationData', {validation_in validation_out}, "ValidationPatience", 15, ...
     'OutputNetwork', 'best-validation-loss'); %training specifics

net = trainNetwork(traininput, trainoutput, layers, options); %training
```

In MATLAB, the training of DL models is handled by the functions `trainingOptions` and `trainNetwork`. The first one is used to implement the desired options and parameters for the training, while the second one is responsible for the actual training. MATLAB allows for the live tracking of the training process, displaying the general loss and the validation loss for every epoch, as well as other useful training parameters, using the 'Verbose' option.

Epoch	Iteration	Time Elapsed (hh:mm:ss)	Mini-batch RMSE	Validation RMSE	Mini-batch Loss	Validation Loss	Base Learning Rate
1	1	00:00:00	1.55	0.95	1.1994	0.4514	0.0050
1	50	00:00:01	0.21	0.20	0.0230	0.0204	0.0050
1	100	00:00:02	0.17	0.18	0.0143	0.0170	0.0050
1	150	00:00:02	0.17	0.16	0.0148	0.0134	0.0050
1	200	00:00:03	0.19	0.17	0.0171	0.0146	0.0050
1	250	00:00:04	0.15	0.15	0.0106	0.0118	0.0050
1	300	00:00:05	0.16	0.16	0.0124	0.0121	0.0050
1	350	00:00:06	0.14	0.15	0.0094	0.0118	0.0050

Figure 6.12. Training process as displayed on MATLAB Command Window.

6.4.5 Testing

The testing is done by using the trained model to make predictions on the testing set. This set must be entirely constituted by observation that the model hasn't seen during the training process, to prevent bias.

The percentage error was evaluated by comparing the prediction with the smoothed signal of the forces, using a moving average filter. This was done because the model was trained on a wide array of data (approximately 600000 observations) with the natural gaussian noise derived by the acquisition source. As explained in chapter 3, noise is actually beneficial for the training of DL models because it allows the network to be able to learn to discriminate between the actual value of the prediction and its noise component.

$$F_{measured} = F_{real} + \varepsilon$$

This way, the network will only predict an approximation of F_{real} and not ε, the noise component.

In other applications, gaussian noise is introduced to the training data for this very purpose. However, here it wasn't necessary because noise was already present due to the vibration and micro adjustments of the actuators during the loading procedure.

The percentage error was also evaluated only for $F_1 > 250N$ and $|F_2| > 25N$. This was done because the initial phases of the loading procedure use very small loads, and a misclassification error of few newtons can lead to a great rise in the percentage error.

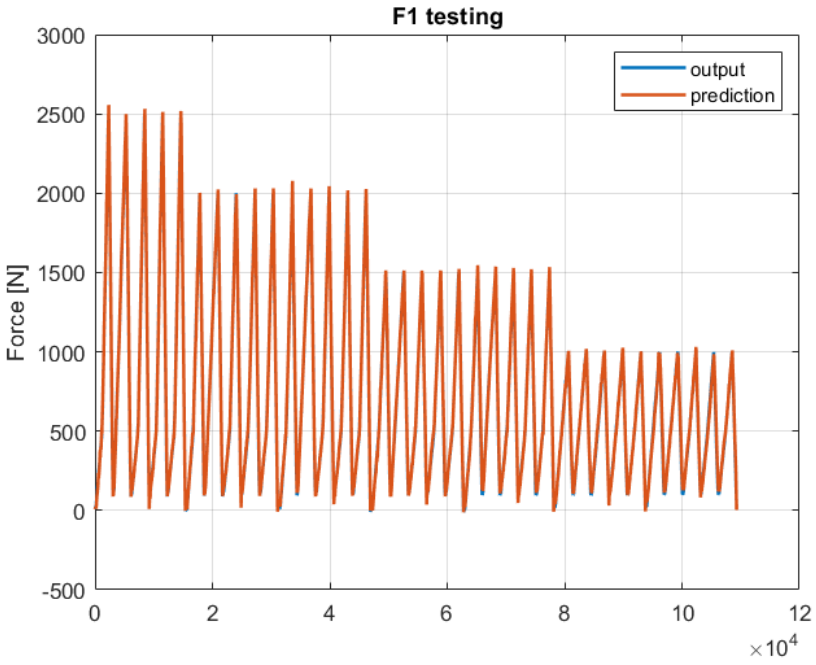
Since the dataset was standardized, the predictions were rescaled to match the original scale of the testing output.

Method 1

The first model, trained with method 1 (without cross validation), was able to achieve the precision of prediction reported in table 6.3.

	$F_1 > 250N$	**$\|F_2\| > 25N$**
Average % error	1.3	3.1

Table 6.3. Average percentage error of the model trained with method 1.



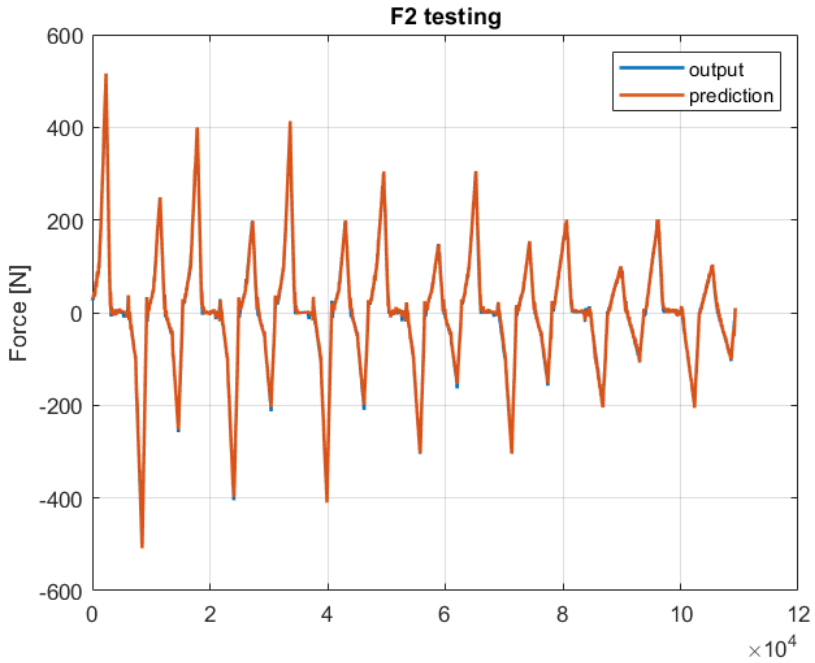


Figure 6.13. Testing results of F_1 (top) and F_2 (bottom). Qualitatively, prediction and output are almost perfectly matching.

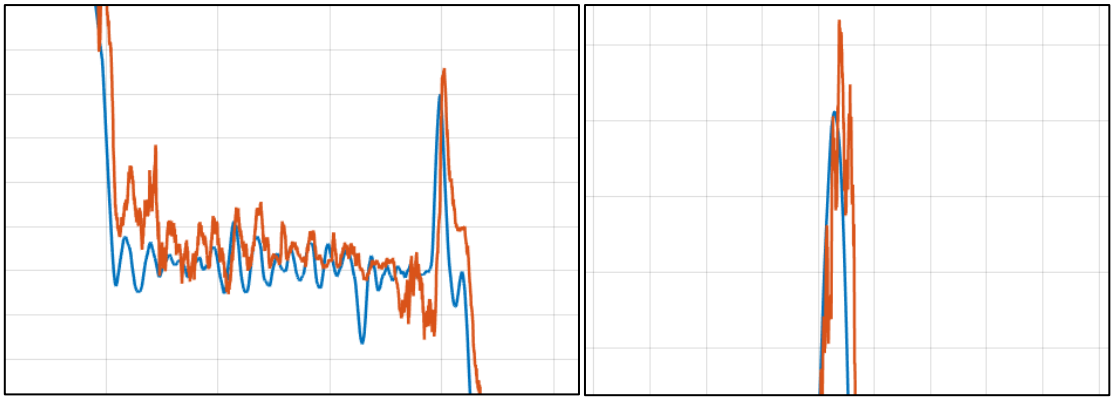


Figure 6.14. Details of $F_2 = 0$ (left) and F_1 peak (right). The reason why the prediction has a higher value than the output on the peaks is because the moving average filter used on the output to smooth the data cuts on the peak values.

Given these results, it is now easy to see that the average percentage errors decrease drastically when excluding small forces. This is especially true for F_2, which is observed to be more difficult to predict. Overall, both predictions have a percentage error inferior to 5%: while the model can predict forces very well, an error so low could also mean it overfits over the angles it was trained over since the validation set and the training set have homogenous observations.

Method 2

The model, with the same specifics as in method 1 except for L2 regularization factor, was tested following a k-fold cross validation on every angle (excluding the boundary ones). As previously explained, this was done to verify the consistency of prediction over untrained angles. A regularization factor of 0.001 was added to improve the generalization capabilities and to prevent overfitting.

The results in terms of percentage error are reported in table 6.4.

	0°	**7.5°**	**-7.5°**	**15°**	**-15°**
F_1 > 250N	6.6	5.6	4.9	4.8	7.2
$\|F_2\|$ > 50N	18.5	16.0	10.3	12.7	16.8

Table 6.4. Average percentage errors for k-fold cross validation sets. The angles on the columns refer to the holdout set over which the network is tested.

Expectedly, the error is higher than in the set trained with method 1 because the network is tested over untrained angles. Though, considering the limited number of angles, the error values for each training are still reasonably low: the average errors never exceed 10% in F_1 and 20% in F_2.

6.5 VALIDATION

The model was validated over the steps of OS1 analyzed in the previous chapter. Though, since the markerset adopted during that session is different from the one the network was trained over, the model was adapted to work with only four markers:

- FP1 (proximal tip of the foot)
- FP2 (center of the reference system)
- FD1 (distal tip of the foot)
- FD2 (behind FD1)

FP2, being the center of the reference system, has null coordinates, so the actual markers used are three. As to be expected, the network trained over only these three markers performs worse, with errors of 6.2% for F_1 > 250N and 10.5% for $|F_2|$ > 25N. For small forces, the error is well over 100% for both. This is to attribute to the lack of the necessary markers.

The kinetic and kinematic data acquired during OS1 were transported to the clamp reference frame, which rotates during the gait motion by an angle is equal to θ_{SH}. The results are reported in figure 6.8.

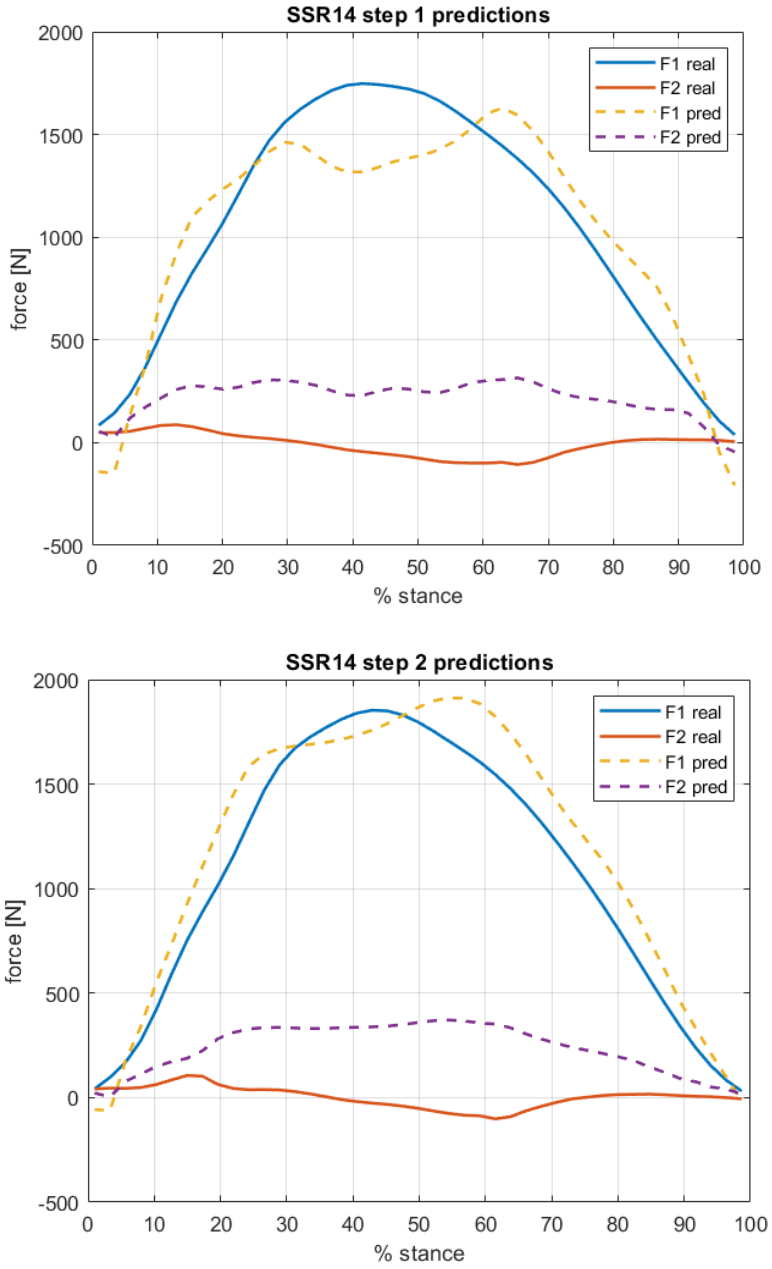


Figure 6.15. Clamp reaction forces predictions for step 1 (top) and step 2 (bottom) of OS1 SSR14.

The markers' coordinates were standardized with respect to the mean and standard deviation used for the training set.

	Step 1	**Step 2**
Average % error F_1 > 250N	35.6	16.3
Average % error $\|F_2\|$ > 25N	68.6	74.8

Table 6.5. Average percentage error of prediction against real value of the force.

6.5.1 Discussion

As can be seen in figure 6.8 and table 6.5, the predictions are imprecise and unreliable, with a very high mean percentage error. Still, the shape of the curve for F_1 is partially correct, which is a sign that the model does not overfit on the trained angles and have good generalization capabilities.

There are a few reasons why the model is so unreliable. The first one is the presence of only three markers, with one of which (FP1) having almost constant coordinates throughout the motion, and thus not contributing much to the predictions. The small number of markers makes it difficult for the network to learn and, consequently, to make accurate predictions. In all likelihood, the full model (with 19 inputs instead of 5) would have shown better performance.

Another factor that may be a cause of the imprecision of these predictions is the fact that the model was trained over constant values of ρ, being 0, ±0.1 and ±0.2, making it difficult to generalize on different force configurations. A more varied set of data would have been beneficial for the generalization capabilities of the model.

For F_1, the prediction is more accurate. Qualitatively, the function follows a "bell curve" except for the peak, where it changes concavity. This is due to the fact that the input data, being the ones obtained from the high frequency cameras as described in chapter 5, had some inaccuracies in the region of maximum compression of the foot.

Regarding F_2, the prediction is very inaccurate. Other than the reasons previously discussed, another cause may be that the forces to predict are small, and the model can't predict them accurately even for data homogeneous to the training set.

Given all these considerations, a new procedure was studied to build an improved model. Said model was built and tested on the target steps of OS3 reported in chapter 5. The process and its results are described in the next chapter.

Chapter 7: Construction of an improved NN model

7.1 STUDY OF A NEW LOADING PROCEDURE

In order to achieve a higher precision of prediction, a new network was built trying to solve the problems of the first prototype. The new model was built and trained over data collected from the Ottobock Runner Standard CAT 3.5 prosthesis that Ambra Sabatini used during OS3 session, using the same markerset (reported in chapter 5).

The first step in building the model was to rethink the loading procedures, to address the problem of having too few values of ρ. Moreover, the study was done focusing on the values the forces take *in vivo*. The result is a procedure that reproduces a neighborhood of possible values of a real step, with ρ assuming continuous values.

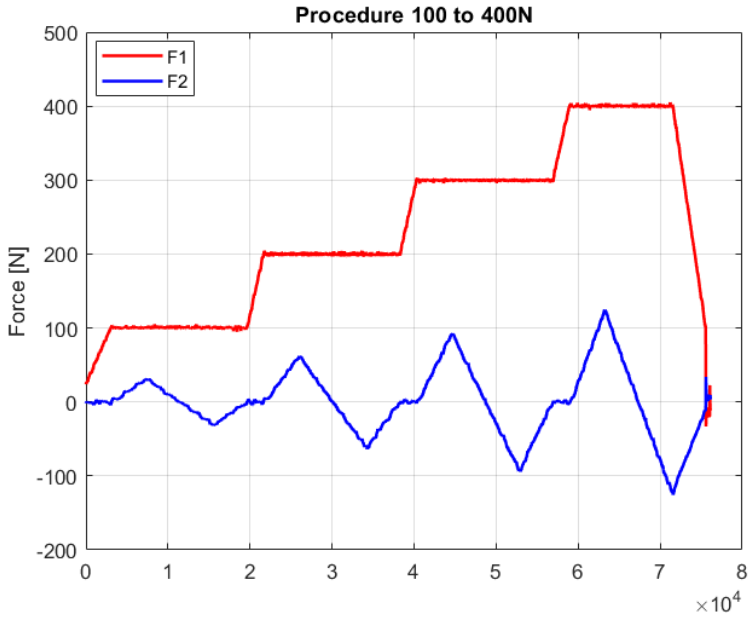


Figure 7.1. Graphic visualization of the new procedure. The procedure from 100 to 400N was reported, for ease of visualization.

Every procedure was studied specifically for one or more angles, with maximum range of force F_1 of 800N.

Here are reported the steps of the procedure in the range 1400-2200N:

1) Linear load of $F_1 = 1400\text{N}$, F_2 hold at 0N (60s)
2) Linear load of $F_2 = \rho_{max} F_1, F_1$ *hold at* $1400N$ (40s)
3) Hold for 2s
4) Linear load of $\mathrm{F_2} = -\rho_{\max}\mathrm{F_1}$, $\mathrm{F_1}$ hold at 1400N (80s)
5) Hold for 2s
6) Linear unload of $F_2 = 0N$, $\mathrm{F_1}$ hold at 1400N (40s)
7) Linear load of $F_1 = 1500N, F_2$ *hold at* $0N$ (20s)

The procedure then repeats steps 2, 3, 4, 5, and 6 iteratively with $\Delta F_1 = 100N$ for every step until it completes the steps at 2200N, after which both cylinders are unloaded to 0N. After that, a procedure with constant values of ρ (±0.2, ±0.1) is launched. $\rho = 0$ is not needed since it is already present in the first procedure: when F_1 is incremented, F_2 is always null.

Each of the new procedures was split in two parts, one that covered the first four steps of ΔF_1 and the other one for the last five. This was done for the acquisition process to be more manageable and to have an easier debugging in case of errors. Each half of the procedure lasted approximately 15 minutes, while the last part (with ρ kept constant) lasted around 12 minutes.

The values of ρ_{max} were chosen based on the real values that ρ assumes during the steps of OS3, with a reasonable margin to cover possible outliers. The values of ρ_{max} and the ranges of F_1 for each angle are reported in table 7.1, as well as the maximum load used during the standard procedures at ρ_{cost}.

	0°	**±7.5°**	**±11.25°**	**±15°**	**±18.75°**	**±22.5°**
$\boldsymbol{\rho_{max}}$	0.2	0.2	0.2	0.2	0.3	0.3
F_1 range for ρ(t)	1400-2200	1400-2200	1000-18000	1000-1800	500-1300	100-800
F_{1max} at $\boldsymbol{\rho_{cost}}$	2000	2000	2000	2000	1500	1000

Table 7.1. Significant parameters of the procedures divided by angle.

For these tests, a higher number of angles was considered. It was particularly important to have at least one more angle between 15° and 22.5°, since it's a region where, for forces acquired during *in vivo* tests, the slope of the F_1 curve becomes progressively more nonlinear. Thus, two

other more angles were tested, being 11.5° and 18.5°. They were reproduced with a flange presenting a hole in the shape of a semicircle. The calibration of the angle was done using a laser caliber.

Another change made for these tests is the value of the maximum force used at ±15° and ±7.5°. Specifically, the forces used for these angles are higher than the ones resulting when computed with the formula reported in chapter 6. This choice is justified by the values that the force assumes during the stride: at said angles, the values computed with the formula are lower than the ones found from *in vivo* data. Since the model must necessarily be trained over the boundary values to make accurate predictions, the force values were risen.

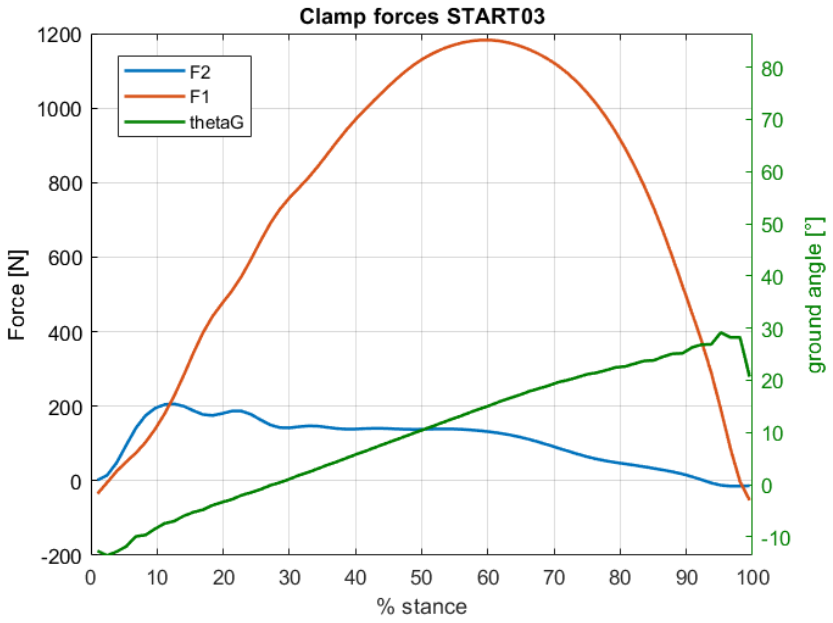


Figure 7.2. Clamp forces plotted with θ_G.

7.2 ACQUISTION OF THE TRAINING DATA

The training data were acquired from several bench tests using the newly studied loading procedure. As in the previous test session, the kinematic data were acquired via optical motion capture; the forces were acquired with a 6 axial load cell connected to the DTS Slice Micro, using the same system described in chapter 5. The RPF used is the Ottobock Runner Standard that was used during OS3.

7.2.1 Markerset

The placement of the markers was done to reproduce as accurately as possible the markerset used during OS3 session. The reconstruction of the markers' coordinates on the RPF was done using the Vicon Nexus software to extract the relative positions. The markers with the same positions on the sagittal plane were substituted with a single marker to ease the tracking and the acquisition.

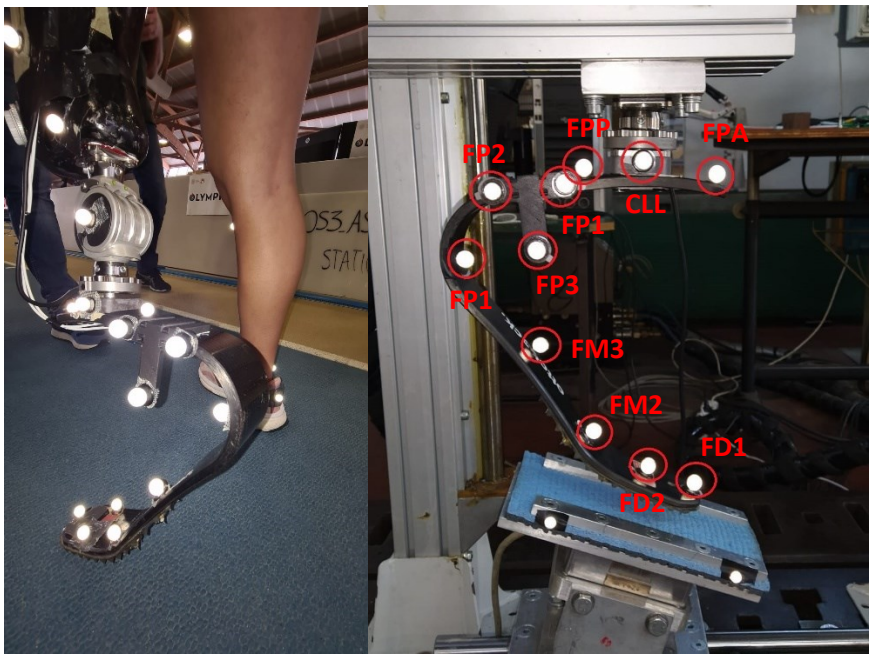


Figure 7.3. Markered Ottobock Runner Standard CAT 3.5 during OS3 (left) and mounted on Colossus (right).

A total of 10 markers was placed on the foot:

- FPA: proximal tip of the foot.
- CLL: center of the clamp reference system in the sagittal plane.
- FPP: behind the clamp.
- FP1: first marker of the triad.
- FP2: second marker of the triad.
- FP3: third marker of the triad.
- FM1: main curvature of the blade.
- FM2: end of curvature and beginning of the distal region.
- FM3: approximately in the middle of the region between the two curvatures.
- FD2: center of the distal region.
- FD1: distal tip of the foot.

Two markers were placed on the slab where the sole of the foot contacts, in order to be able to track the possible variability of θ_G during the whole loading procedure. Moreover, it allowed to have more accurate angles for the training of the model.

As for the positioning of the foot on the bench, its alignment was adjusted with a spirt level on the sagittal and frontal plane. the tap was set to 5 cm, using the same value of OS3.

7.2.2 Acquisition of kinematic data

Similarly to what was done for the previous set of tests, the kinematic data were acquired placing 5 Vicon Bonita cameras around the testing machine. Four were placed on the sagittal plane and 1 on the frontal plane. The operations carried out to prepare the Vicon acquisition system were done following the standard procedure, reported in section 6.3.2.

The acquisition frequency was lowered to 50Hz in order to have more manageable data, since the procedures to be tested were much longer than in the previous session.

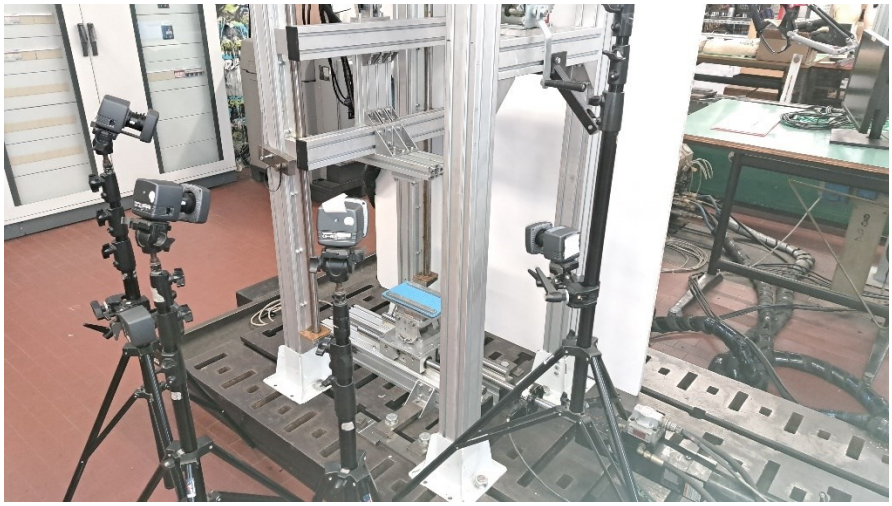


Figure 7.4. Position of the cameras around the test bench.

7.2.3 Acquisition of dynamic data

The dynamic data were acquired using the same load cell and the same system used in OS3. The load cell was in fact placed in series with the other constituting elements of the pylon, as described in chapter 5.

The procedure to acquire data with DTS Slice Nano is the following:

1) Connection of the base slice to controller and to power supply;

2) Connection of the first and second bridge slices, both holding 3 channels, to the computer that manages the DTS SLICEware software;
3) Setting of the frame rate (here set 50Hz to match the one of the MOCAP system) and of the region of interest (ROI) on the software interface;
4) Powering of the DTS Slice Nano by turning on the controller;
5) Arming of the acquisition system is the software interface;
6) Start of the acquisition with the controller;
7) When the acquisition is finished, the “start” button on the controller is turned off and the red button is pushed to save the data.

The resulting procedure for the acquisition of the data and the launch of the loading test is then:

1) Setting of the angle θ_G;
2) Arming of the DTS Slice Nano;
3) Positioning of C2 in the appropriate spot (50mm for low angles);
4) Start of the acquisition for Vicon Nexus and DTS;
5) Imposition of a preload of 25N on C1;
6) Start of the loading procedure.

When the loading procedure is finished, both acquisitions are stopped. The MOCAP data are processed by the core processor, labelled, and saved in .c3d files; the dynamic data are saved in .CHN and .DTS files, which can be easily converted in .csv files in the SLICEware interface.

7.3 COSTRUCTION OF THE NEURAL NETWORK

The network was built in MATLAB using the same toolboxes of the prototype. Since the model studied to address the problems of the prototype did not give optimal results, other models were tested to investigate the possible reasons of its poor performance. The most interesting and best performing models are now reported; the value of percentage error for every model, evaluated as explained in chapter 6, is reported at the end of this section in table 7.2.

7.3.1 Model 1

The data were first extracted in the same way reported in the previous chapter; then they were preprocessed. Similarly to the prototype model, the first model was built and trained over the markers' coordinates and the ground angle θ_G. The latter was computed using the coordinates of the markers placed on the ground slab, taking into account the possible sliding.

The preprocessing was done following these steps:

1) Setting of the coordinates in the clamp reference frame using the marker CLL as its center.
2) Interpolation of the data. Since the matrix containing the coordinates of the markers and the one containing the forces were not perfectly synchronous due to the limitations of the acquisition setup, the least numerous in every set was thickened using the `interpft` function. MATLAB offers several functions for interpolation: however, this function was the one that allowed to achieve the highest precision of prediction.
3) Scaling of the data. As explained in the previous chapter (see section 6.4.2), in DL applications it is good practice to scale the input and output data for a better training process. The first attempt was done by standardizing the data with null mean and unitary variance, but this gave terrible results during the training: the loss function converged on very high values (around 0.1), giving rise to severe underfitting on both validation and testing sets. The problem persisted even in tests carried out on high numbers of layers and neurons, which would normally be prone to overfitting the training data. It is not clear why this happened; however, the problem was solved by normalizing all the data in the [-1, 1] interval.

The whole data set was then split into three sets (training, testing, and validation) using the `datasample` function.

The most performing architecture was the same used for the prototype, constituted by 3 hidden layers with ReLU as the activation function and 128 neurons per layer.

```
neurons = 128;
x = 21; % input variables
y = 2; %output variables
layers = [
    featureInputLayer(x, "Name", "Input")
    fullyConnectedLayer(neurons, "Name", "FC1")
    reluLayer("Name", "act1")
    fullyConnectedLayer(neurons, "Name", "FC2")
    reluLayer("Name", "act2")
    fullyConnectedLayer(neurons, "Name", "FC3")
    reluLayer("Name", "act3")
    fullyConnectedLayer(y, "Name", "FC4")
    regressionLayer("Name", "reg")
    ];
```

The configuration of parameters chosen, able to achieve the lowest error, is the following:

Optimizer: adam

Initial learn rate: 0.005

Maximum epochs: 15

Size of mini batch: 256

Learn rate drop factor: 0.1

Learn rate drop period: 3

Validation patience: 20

L2 regularization factor: 0.001

The training process was always stopped by the validation criterion for all the models considered, and never reached the maximum number of epochs.

```
options = trainingOptions("adam", "MaxEpochs", 15,  'InitialLearnRate',0.005,...
    'Verbose',true, 'MiniBatchSize', 256, "Shuffle", "every-epoch",...
     'LearnRateSchedule', 'piecewise', 'LearnRateDropPeriod', 3,...
    'LearnRateDropFactor',0.1, 'L2Regularization', 0.001, ...
       'ValidationData', {validin validout}, "ValidationPatience", 20, ...
       'OutputNetwork', 'best-validation-loss'); %training specifics
```

The network was trained and then its performance was evaluated on the testing set using the mean percentage error. Given the premises, the performance was underwhelming: the predictions of $F_1 > 250N$ had an error of ~13%, while for $|F_2| > 25N$ the error was ~41%.

Several attempts of tweaking the network's parameters have been made to both decrease the convergence value of the loss function on the training set and of the percentage error on the testing set, all with unsatisfactory results. Increasing the number of neurons and layers, which is the most straightforward approach to increase performance, made the training slower but did not lower the training and the validation losses. Other attempts concerned the tuning of the hyperparameters, the use of SGD and SGDM as optimizers, the use of different activation function, and the simplification of the model.

The possible reasons why this network was outperformed by the prototype will be discussed later; now, another attempt made using a more heterogenous training set is reported.

7.3.2 Model 2: introduction of new variables

The second model was built to understand if considering a training set with more variables could improve the performance. This was done calculating the by calculating the Euclidean distance of every pair of markers and adding it to the training set as inductive bias.

The combination of all the couples of markers can be calculated using the following formula:

$$C(n,k) = \frac{n!}{k!\,(n-k)!}$$

Having n as the number of markers (10) and k as the number of markers in each group (2). The result is 45 more variables, being all the distances. These variables were all added to the dataset. The reason for this choice lies in the nature of DL applications: by design, a Multilayer Perceptron can only approximate functions. The hypothesis made here was that it could be beneficial for the training process to feed the network the exact value of nonlinear functions that might be helpful for the approximation of the overall function that links the coordinates of the markers to the forces acting on the RPF.

The model was trained both by standardizing and normalizing the variables, giving better results when applying standardization. This result is interesting when compared to the previous case, where standardization of the data led the model to converge on very high values of training loss. The architecture and training specifics were kept the same as in the first model, and they were verified to be the most performing with a set of tests.

The performance of this model was better than the one of the first one, but still not optimal: the percentage error was 11% for $F_1 > 250N$ and 31% for $|F_2| > 25N$.

The introduction of new variables was beneficial but did not give satisfactory results. The next step attempted was to try to move in the opposite direction and investigate if a substantial simplification of the training set, replicating the one used in the prototype, could lead to better results.

7.3.3 Model 3: simplification of the training set

The third model was studied to understand if the data used were too complex and could lead to ambiguity in the prediction. Thus, the training data were simplified to match the level of complexity of the ones used for the prototype:

- Only data from tests at constant values of ρ were used, ignoring the ones acquired from the new procedures;
- A lower number of markers was used.

Several tests have been carried out, using the same training specifics as the other models previously described. The set of markers able to achieve the best performance is constituted by FPA, FPP, FM1, FD2, and FD1 (reported in figure 7.5 in top to bottom order), although other configurations gave results just slightly worse.

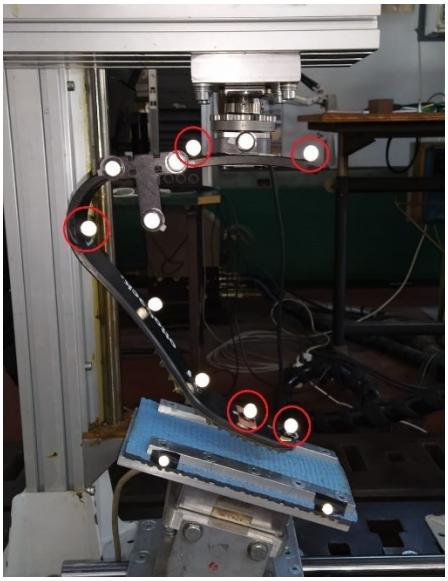


Figure 7.4. Best performing set of the markers for model 3.

As for model 2, the data were standardized with null mean and unitary variance without stumbling in the problem seen in the first model. The performance of this model was significantly better than the previous two: the average percentage error was equal or inferior to 10% for both forces, being specifically 5% for $F_1 > 250N$ and 10% for $|F_2| > 25N$.

An additional test was carried out, using the data acquired from the new procedures with the same simplified set of markers. Here, the model returned to perform poorly with errors of 10% and 27% respectively. This drop in performance clearly indicates that the reason for the poor performance of the other models lies in the data collected from the new procedures: the possible issues will be discussed in the next section.

A summary of the performances of the various models discussed is proposed in table 7.2, with the inclusion of other tests carried out to completely unwrap the doubts arose with the first two models. In particular, two tests are reported where the number of markers was inferior to model

3, with the sole use of 2 (FM1, FD1) or 3 (FM1, FD1, FD2) markers. This was done following the geometric reasoning of the preliminary tests (see section 6.2.2), to verify:

- The minimum number of markers. During the initial trial and error experiments, it was presumed that the variation of the geometry of the foot could be expressed through functions derived by the triangle with vertices in CLL-FM1-FD1. Here, it is clear that an higher number of markers improves the performance of the model, suggesting the three markers considered are not sufficient for a full expression of the geometry of the prosthesis.
- The significance of the marker FD2 in the estimation of the forces. FD2 is the marker placed in the distal region behind FD1, which is placed on the tip. It appears that the presence of FD2 makes a significant difference, improving the performance of the model especially in the estimation of F_1. This could be justified by the fact that its presence allows to better characterize the orientation of the distal region.

It is also worth noticing that the inclusion of the Euclidean distances for the third model increases the performance just slightly.

Model	**Training set characteristics**	**Markers**	**Average % error $F_1 > 250N$**	**Average % error $\lvert F_2 \rvert > 25N$**
1	ρ = cost ρ = ρ(t)	Full set	13	40
2	ρ = cost ρ = ρ(t) Inclusion of Euclidean distances	Full set	11	31
3	ρ = cost	FPA, FPP, FM1 FD1, FD2	5	10
3, test 1	ρ = cost	FM1, FD1, FD2	7	11
3, test 2	ρ = cost	FM1, FD1	11	15
3, test 3	ρ = cost ρ = ρ(t)	FPA, FPP, FM1 FD1, FD2	10	27
3, test 4	ρ = cost Inclusion of Euclidean distances	FPA, FPP, FM1 FD1, FD2	5	9

Table 7.2. Summary of the performances of the models.

7.3.4 Discussion

The tests carried out on the previous models gave rise to interesting considerations. Several reasons were found for the less complex model to be the best performing, and for the data collected with the new procedures to give such underwhelming results.

Correlating function: the first issue to address lies in the fundaments of the problem analyzed in this work, being the function C which correlates the geometric configuration of the foot, expressed through the coordinates of the markers, to the forces acting on it.

$$F = (F_1, F_2) = C(m_1, m_2, \dots, m_n)$$

Said function is what the Neural Network seeks to approximate. During all the analysis carried out in this paper, one key assumption was made, being that the correlating function was bijective. This means that every geometric configuration of the foot corresponds to one and one only one couple of forces:

$$(F_1, F_2) \leftrightarrow (m_1, m_2, \dots, m_n)$$

The results obtained with the construction of these models lead to consider that there may be ambiguity when considering variable values of ρ, having two or more couples of forces resulting from the same geometric configuration of the foot, thus disproving the assumption of bijectivity. Though, these considerations cannot be fully proven from the information gathered in this paper without further investigations.

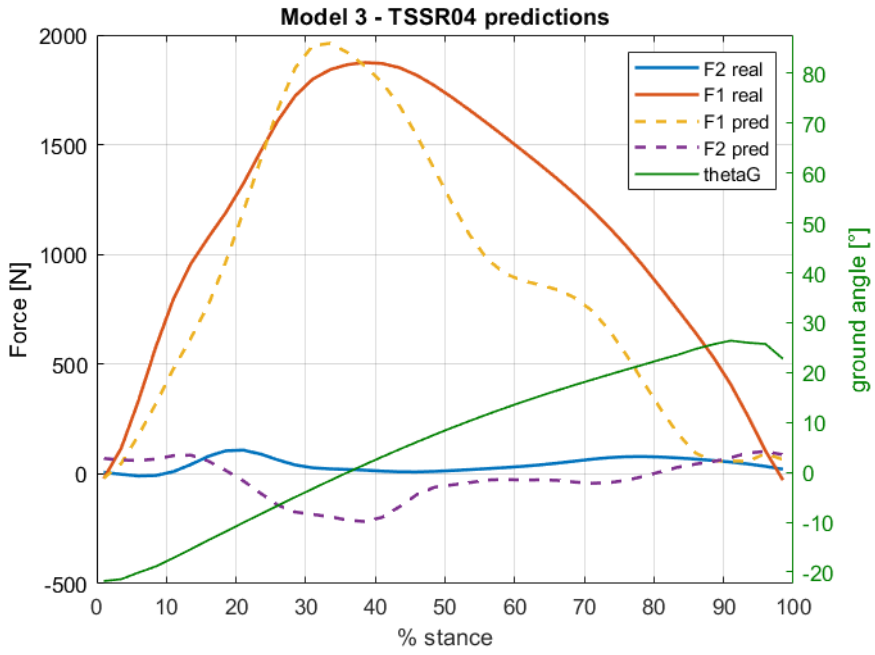
Markers protocol: the correlating function C is directly dependent from the set of markers used. This means that the effectivity of the network in the approximation of C depends on how well the markers can represent the geometric configuration of the foot at a given time. The results of the tests performed on the third model, along with the results of the validation test of the prototype, lead to consider that there is an optimal number and placement of markers that can describe the cinematic behavior of the prosthesis with good approximation and with little ambiguity, and that it directly influences the performance of the network.

Model type: another issue to consider is the type of model used. For all the tests the model was a MLP with variable number of neurons and layers; the best performance was always given by three hidden layers with 128 neurons per layer. It is possible that these configurations of MLP

may not be best suited for the problem when considering more complex datasets. More refined ML models, such as the ones that can be built with the XGBoost library, may perform better. Moreover, the implementation of a more elaborated architecture could also be beneficial.

7.4 VALIDATION

Being the best performing model, model 3 was validated over the same three steps of OS3 analyzed in chapter 5. The markers considered were the ones that allowed the network to achieve the optimal performance during the testing phase (reported in figure 7.4). The markers' coordinates were standardized with respect to the mean and standard deviation used for the training set.



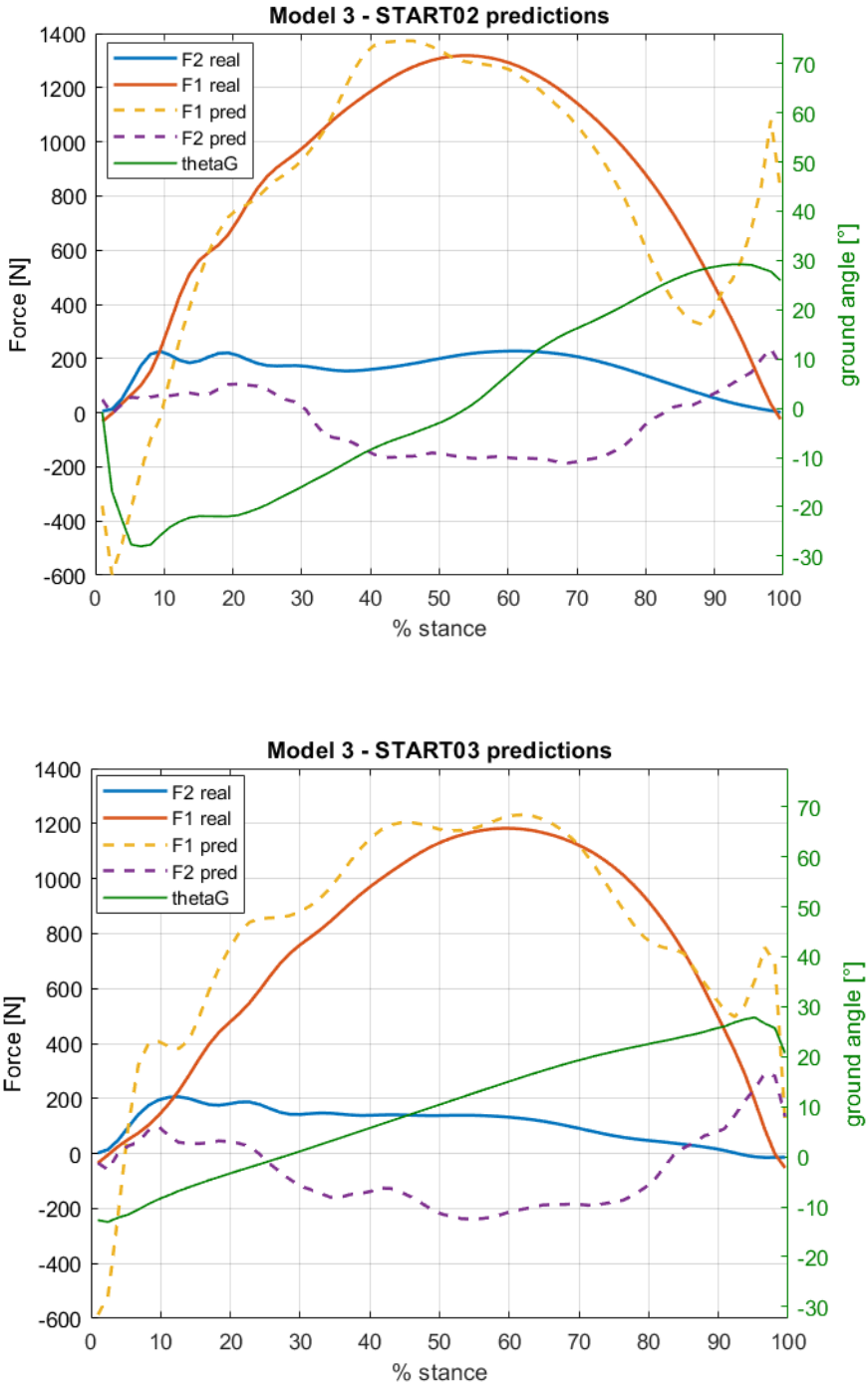


Figure 7.5. Clamp reaction forces predictions for TSSR04 (top), START02 (middle), and START03 (bottom) of OS3 session.

Overall, the prediction tends to be more accurate for F_1, particularly in the region of maximum load. Observing the graphs referring to the START tests, there are regions where the predictions can clearly reproduce the trend of the real curve, indicating good sensibility to small changes. Though, in the case of F_2, the value is often noticeably off. This is a shared characteristic with

the prototype, which reflects the difficulty in the prediction of F_2 even during the testing of the model.

It is important to notice that the predictions are most unreliable in the regions where they refer to angles outside of the boundary ones, over which the model has not been trained. These regions, corresponding to $|\theta_G| > 22.5$, have not been considered in the computation of the error, since they do not reflect the capabilities of the model. There are also areas where the real value of the force exceeds the maximum one of the loading tests for the corresponding angle, again leading the network to predict outside its training boundaries.

In the region of peak vertical force the curves match for START tests, while for TSSR04 the prediction seems to drop in accuracy in the area corresponding to the propulsive phase. It is not clear why this happens; the reason could lie in imprecisions of the video data.

In table 7.3 the values of the average percentage error are reported for all three steps, as well as the ones of OS1 as means of confrontation.

	OS1		**OS3**		
	Step 1	**Step 2**	**TSSR04**	**START02**	**START03**
Average % error F_1 > 250N	35.6	16.3	21.8	10.0	16.4
Average % error $\|F_2\|$ > 25N	68.6	74.8	114.3	140.7	172.6

Table 7.3. Average percentage error of prediction against real value of the force for OS1 and OS3.

It is interesting to notice that the error for F_2 is lower for the steps of OS1, even though the two models had a similar testing performance. This can be justified by the fact that the prototype was trained over sets of tests with ρ=0, which were not present for model 3 (since they were incorporated in the new procedures, not included for the training of model 3). Still, the error is very high for both models and the prediction for F_2 are completely unreliable.

As for F_1, model 3 just slightly exceeds a 20% error on TSSR04, while for the prototype it is well over 30% for step 1 and close to 15% for step 2. This could indicate that the third model predicts the vertical force better, but statistically motivated assessments cannot be made because the number of samples is too low.

7.5 FINAL CONSIDERATIONS AND POSSIBLE IMPROVEMENTS

The model described in this chapter was built with the intention of improving the first prototype. It is not entirely clear why the prototype had a significantly better performance for F_2 and a worse one for F_1. Anyway, both models could approximate the shape of the real force curves, although not always with exact values. This indicates good sensitivity and generalization capabilities of the models.

Moreover, the training of the latest model using the new procedures was beneficial to shed light on the correlation between the geometric configuration of the RPF and the corresponding forces, indicating a possible non-bijectivity of the correlating function. It would be ideal to perform an in-depth statistical analysis to furtherly investigate this matter, as well as studying a protocol for the optimal placement of markers, to eliminate possible ambiguity and still well represent the geometric configuration of the foot.

As for the overall approach followed, it can be concluded that AI can be used to address the problem studied in this thesis. Still, there is a need for major improvements, especially with the experimentation of different and more powerful ML models. Moreover, training data with wider boundaries would significantly improve performance.

Conclusions

The work described in this thesis was carried out to answer two different research questions.

1) Are kinematic data obtained from 2D videos a good approximation of MOCAP data when analyzing a RPF during running?
2) Is it possible to derivate the forces acting on a RPF solely from its geometric configuration?

The first question was addressed by using data collected from two running sessions, OS1 and OS3, with two elite paralympic athletes. The same steps were captured using optical motion capture and video cameras, and the confrontation was carried out by evaluating the difference of the tip-to-tip distance over the whole step. It was observed that the percentage error between the two acquisition systems never exceeds 3%, although the video data are more susceptible to actions on the frontal plane, whose magnitude cannot be fully evaluated. Moreover, it was observed that setting the scene with proper illumination and minimizing distortion effects makes a drastic difference in the tracking of video data, leading to a much more accurate process.

For the second question, the problem was first addressed with analytical and statistical considerations. Kinematic and dynamic data were acquired from a set of *in vitro* tests, carried out on hydraulic test bench Colossus, and analyzed from a geometrical standpoint. Since this approach didn't lead to satisfactory results, the problem was formalized using Deep Learning: a first Neural Network was built, in the form of a Multilayer Perceptron, to correlate kinematic and dynamic data in the form of a regression problem. The model was then trained and tested over data collected from a second set of tests, performed with seven different angles and five force ratios. This model served as a prototype to understand the optimal characteristics, in terms of architecture and training parameters, to achieve a low error without incurring in overfitting. The best performing model achieved a percentage error of 1% for the vertical force and 3% for the horizontal one (with respect to a clamp reference system), though k-fold cross validation showed that the error on untrained angles was higher for both forces. It was then validated on two steps taken from OS1, using a lower number of markers. Its performance was underwhelming, achieving a precision of prediction not inferior to 15% for the vertical force and to 65% to the horizontal one.

A second model was then built to try to address the problems of the first one. A new set of testing procedures, studied to collect more varied data, was carried out, with the inclusion of the standard procedures used for the prototype. The collected data were used to train a new model: surprisingly, the data obtained from the new procedures were responsible for a significant drop in precision on the testing set, achieving errors of 13% and 40% for the two forces respectively. To investigate these results, some experiments were performed. It was concluded that there is a possibility that the new data introduced ambiguity in the training set, having the same geometrical configurations of the foot corresponding to different pairs of forces. The drop in performance could also depend by the set of markers and by the type of model.

The best performing model, only trained over the standard tests, was then validated on three steps of OS3. Its performance was acceptable for the estimation of the vertical force (with an error no greater than 20%) but very poor for the horizontal one (with errors well over 100%). It was not fully understood why it was outperformed by the prototype, but still its construction gave useful insight on the correlation between the geometry of the foot and the forces acting on it.

In the end, it was concluded that an AI approach can be used for the estimation of forces, but it needs major improvements. Future goals should focus on deeply studying the problem from a statistical standpoint and experimenting with different ML models.

Bibliography

Beck, O. N., Taboga, P., & Grabowski, A. M. (2016). Characterizing the Mechanical Properties of Running-Specific Prostheses. *PLOS ONE*, *11*(12), e0168298. https://doi.org/10.1371/journal.pone.0168298

Bento, C. (2021, 21 September). *Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis*. Medium. https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141

Cognolato S. (2021). *Study of emergent properties and representations in the discovery of an elementary calculation system through deep learning* [Master's degree thesis, Department of Information Engineering, University of Padua].

Costa G. (2017). *Design and construction of a multichannel bench test for running specific prostheses* [Master's degree thesis, Department of Industrial Engineering, University of Padua].

Dugan, S. A., & Bhat, K. P. (2005). Biomechanics and Analysis of Running Gait. *Physical Medicine and Rehabilitation Clinics of North America*, *16*(3), 603–621. https://doi.org/10.1016/j.pmr.2005.02.007

Foscan G. (2017). *Structural analysis and functional behaviour of Running Specific Prosthesis during in-vivo field tests and laboratory bench tests* [Master's degree thesis, Department of Industrial Engineering, University of Padua].

Gri A. (2018). *The effect of socket alignment on the running performance of elite Paralympic athletes during indoor and outdoor tests using an instrumented Running Prosthetic Foot* [Master's degree thesis, Department of Industrial Engineering, University of Padua].

Hobara, H. (2014). Running-specific prostheses: The history, mechanics, and controversy. *Journal of the Society of Biomechanisms*, *38*(2), 105–110. https://doi.org/10.3951/sobim.38.105

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8

Kingma, D. P., & Ba, J. (2017). *Adam: A Method for Stochastic Optimization* (arXiv:1412.6980). arXiv. http://arxiv.org/abs/1412.6980

Martellato A. (2022). *Numerical analysis of the structural behaviour of Running Prosthetic Feet and validation against paralympic athletes treadmill running data* [Master's degree thesis, Department of Industrial Engineering, University of Padua].

Migliore, Petrone, Hobara, Nagahara, Miyashiro, Costa, Gri, & Cutti. (2021). *Innovative alignment of sprinting prostheses for persons with transfemoral amputation: Exploratory study on a gold medal Paralympic athlete*.

Petrone, N., Costa, G., Foscan, G., Gri, A., Mazzanti, L., Migliore, G., & Cutti, A. G. (2020). Development of Instrumented Running Prosthetic Feet for the Collection of Track Loads on Elite Athletes. *Sensors*, *20*(20), 5758. https://doi.org/10.3390/s20205758

Razavi, S. (2021). Deep learning, explained: Fundamentals, explainability, and bridgeability to process-based modelling. *Environmental Modelling & Software*, *144*, 105159. https://doi.org/10.1016/j.envsoft.2021.105159

Rumelhart, D. E., Hintont, G. E., & Williams, R. J. (1986). *Learning representations by back-propagating errors*. 4.

Sano, Y., Makimoto, A., Hashizume, S., Murai, A., Kobayashi, Y., Takemura, H., & Hobara, H. (2017). Leg stiffness during sprinting in transfemoral amputees with running-specific prosthesis. *Gait & Posture*, *56*, 65–67. https://doi.org/10.1016/j.gaitpost.2017.04.038

Shepherd, M. K., Gunz, D., Lecomte, C., & Rouse, E. J. (2019). Methods for Describing and Characterizing the Mechanical Behavior of Running-Specific Prosthetic Feet. *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, 892–898. https://doi.org/10.1109/ICORR.2019.8779557